National Technical University of Athens School of Electrical and Computer Engineering Division: Information Transmission Systems and Material Technology

Technologies for the Development of Integrated Oncosimulators

Ph.D. Dissertation

Nikolaos A. Christodoulou

Athens, July 2025

National Technical University of Athens
School of Electrical and Computer Engineering
Division: Information Transmission Systems
and Material Technology

Technologies for the Development of Integrated Oncosimulators

Ph.D. Dissertation

Nikolaos A. Christodoulou

Three-member doctoral scientific committee : Nikolaos K. Uzunoglu, Professor Emeritus NTUA

Georgios S. Stamatakos, Research Professor ICCS – NTUA, Visiting Professor NTUA (2016 – 2019)

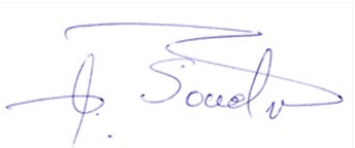
Stefanos D. Kollias, Professor Emeritus NTUA

Approved by the seven-member doctoral scientific committee on the 29th July 2025

Nikolaos Uzunoglu,
Professor Emeritus NTUA

Georgios Stamatakos,
Research Professor ICCS - NTUA,
Visiting Professor NTUA (2016-2019)

Stefanos Kollias,
Professor Emeritus NTUA



Dimitrios Soudris,
Professor NTUA

Nektarios Koziris,
Professor NTUA

Dimitra-Theodora Kaklamani,
Professor NTUA

Panayiotis Frangos,
Professor NTUA

Athens, July 2025

Nikolaos A. Christodoulou
PhD, National Technical University of Athens

Περίληψη

Σκοπός της παρούσας διδακτορικής διατριβής είναι η ανάπτυξη και η περιγραφή τεχνολογιών δημιουργίας ολοκληρωμένων Ογκοπροσομοιωτών, δηλαδή ψηφιακών διδύμων (digital twins) στην ογκολογία. Η ανάγκη για μια τέτοια διαδικασία προκύπτει από το γεγονός ότι ένα μοντέλο προσομοίωσης κατά βάση παραμένει ένα κομμάτι λογισμικού, του οποίου η είσοδος και η έξοδος στις αρχικές μορφές του δίνουν μεν την απαραίτητη κλινική πληροφορία προς εξαγωγή συμπερασμάτων, πλην όμως οι μορφές αυτές δεν είναι άμεσα ή εύκολα αναγνώσιμες και κατανοήσιμες είτε από τον ερευνητή, είτε από τον κλινικό ιατρό.

Ως πρόταση για την επίλυση αυτού του σημαντικού ζητήματος που έχει ως απώτερο σκοπό την εφαρμογή επί του κλινικού πεδίου της έννοιας της προσωποποιημένης in silico ιατρικής, προτείνεται μια υπολογιστική υποδομή, της οποίας η βασική λειτουργικότητα επικεντρώνεται στην αποθήκευση των χαρακτηριστικών ενός προσομοιωτικού μοντέλου, συμπεριλαμβανομένων και των σχετικών εκτελέσιμων και συνοδευτικών αρχείων αυτού. Αρχικά παρέχεται μια εισαγωγή στις βασικές αρχές ανάπτυξης και λειτουργικότητας μιας τέτοιας υποδομής, ενώ παρουσιάζονται προτεινόμενες επεκτάσεις για το χειρισμό σημασιολογικών δεδομένων, καθώς και κλινικών ερωτήσεων. Παράλληλα, μέσα από τα αποτελέσματα επιτυχημένων Ευρωπαϊκών ερευνητικών προγραμμάτων παρουσιάζονται σενάρια χρήσης προσομοιωτικών μοντέλων στα οποία μπορεί να συνεισφέρει η εν λόγω υποδομή, ώστε να προωθηθεί η μετάβασή τους στην καθημερινή κλινική πράξη, όπως για παράδειγμα η χρήση των Ογκοπροσομοιωτών σε Συστήματα Υποστήριξης Κλινικών Αποφάσεων.

Στη συνέχεια δίνεται μια πρωτότυπη υλοποίηση της προαναφερθείσας υποδομής. Περιλαμβάνει την περιγραφή των τεχνολογιών και τεχνικών σχεδίασης που χρησιμοποιούνται, τα επιμέρους μέρη στα οποία αυτή διαχωρίζεται και στα σενάρια χρήσης της. Κατόπιν παρουσιάζονται τα αποτελέσματα της χρήσης της υλοποιημένης εφαρμογής μέσα από αυτά τα σενάρια. Τέλος γίνεται μια σύντομη σύνοψη που περιλαμβάνει και μελλοντικές επεκτάσεις.

Λέξεις Κλειδιά

Ογκοπροσομοιωτής, In Silico Ογκολογία, In Silico Ιατρική Αποθετήρια Μοντέλων και Δεδομένων, Κλινική Ερώτηση, Σύστημα Υποστήριξης Κλινικών Αποφάσεων, Σύστημα Υποστήριξης Κλινικών Μελετών

Abstract

The purpose of this doctoral dissertation is to develop and document technologies necessary for integrated Oncosimulators. i.e. digital twins in oncology. The need for these processes arises from the fact that a simulation model basically remains a piece of software, whose input and output in their original forms indeed provide the necessary clinical information for drawing conclusions, but in a format not immediately or easily readable and understandable by either the researcher or the clinician.

Within the context of solving this significant issue, which ultimately aims at the clinical application of the concept of personalized in silico medicine, a computational infrastructure is proposed. Its basic functionality focuses on storing the characteristics of a simulation model, including its relevant executable and accompanying files. Initially, an introduction to the basic principles of development and functionality of such an infrastructure is provided, while proposed extensions for handling semantic data and clinical questions are presented. At the same time, through the results of successful European Commission funded research projects, use cases of simulation models to which the aforementioned infrastructure can contribute are presented. These include Oncosimulators and their transition to routine clinical practice in the form of Clinical Decision Support Systems.

Subsequently, an original implementation of the aforementioned infrastructure is provided. This includes a description of the design techniques and technologies used, the individual parts into which it is divided, and its use cases. Furthermore, the results of using the implemented application through these scenarios are presented. Finally, a brief summary, including future extensions is provided.

Key Words

Oncosimulator, In Silico Omncology, In Silico Medicine, Model and Data Repositories, Clinical Question, Clinical Decision Support System, Clinical Study Support System

Acknowledgements

I would like to thank Professor Emeritus Nikolaos Uzunoglu for the great opportunity he offered me to carry out a clinically relevant doctoral thesis at the School of Electrical and Computer Engineering (ECE), National Technical University of Athens (NTUA), as well as for his overview throughout the thesis preparation.

I am indebted to Research Professor Georgios Stamatakos, Director of the In Silico Oncology and In Silico Medicine Group, ECE, NTUA, for introducing me into the formidable world of in silico medicine and especially into in silico oncology and the Oncosimulator digital twins. His invaluable in-depth guidance throughout the various stages of the lengthy thesis preparation process were pivotal to its completion. Furthermore, I am grateful to Georgios for the opportunities he offered me to work under his leadership for several European Commission funded research projects. These projects provided substantial clinical and technical context and input to the present thesis.

I thank Professor Emeritus Stefanos Kollias for his precious and multifaceted support and encouragement throughout the thesis preparation.

Many thanks are also due to the other Members of the Examination Committees i.e. Professor D. Soudris, Professor N. Koziris, Professor P. Frangos and Professor D. Kaklamani for their time demanding participation in the thesis examination. Their questions, comments and feedback were highly constructive.

Special thanks go to Professor Norbert Graf, MD, University of Saarland Germany, for his exemplary and inspirational contributions to in silico paediatric oncology in terms of clinical expertise, clinical vision and clinical translation.

I also duly acknowledge the support of my former colleagues Eleni Georgiadi, Katerina Argyri, Eleni Kolokotroni and Foteini Panagiotidou. Through their rigorous and meticulous work, certain Oncosimulator versions were used as examples for the web application developed in this thesis

Last but not least, I would like to thank my family, who stood by me even during the most difficult phases of this demanding intellectual endeavour.

Table of Contents

Περίληψη....5
Abstract....7
Acknowledgements....9
Table of Contents....10
List of Figures....12
List of Tables....14
Εκτενής Περίληψη....15
Extended Abstract....27
1. Introduction....37
1.1. In Silico Oncology – Definition and main objectives....37
1.2. The Oncosimulator....38
1.3. Main technological challenges towards personalize medicine....41
1.4. State of the Art....43
1.5. Chapter overview....44
1.6. References....45
2. Basic Principles for Model Hosting Infrastructures....50
2.1. Conceptual system overview....50
2.2. Components and Interrelations....50
2.3. The MyHealthAvatar paradigm....55
2.3.1. Implementation stack....55
2.3.2. The Wilms tumor Oncosimulator....55
2.3.3. The breast cancer Oncosimulator....56
2.3.4. Indicative Workflows....57
2.3.5. Legal/ethics/security considerations....58
2.4. References....60
3. Metadata Creation for In Silico Models – A theoretical approach....62
3.1. Components Overview....62
3.1.1. User Roles....62
3.1.2. Entry Points....63
3.1.3. User Communication Components....64
3.1.4. Storage Components....65
3.2. The CHIC Project Paradigm....65
3.3. Suggested expansions....66
3.4. References....66
4. Integration with Clinical Decision Support Systems (CDSS)....70
4.1. CDSS definition....70
4.2. How CDSS Workflows Can Help Personalized Medicine....71
4.3. The Oncosimulator Paradigm: A Foundation for Personalized Oncology....72
4.3.1. Integrating the Oncosimulator with Clinical Decision Support Systems....73
4.3.1.1. The breast cancer Oncosimulator....73
4.3.1.2. The Wilms tumor Oncosimulator....75
4.3.2. Integration workflow and results....75
4.4 References....79
5. Standardization of Clinical Questions....81
5.1. The overall need for standardization....81
5.2. Existing tools and technologies for proper standardization....82
5.3. Implementation Challenges....83

5.4. Conceptual system overview....84
5.5 User workflow....86
5.6 References....87
6. The Hyperion Application....90
6.1. Overview....90
6.1.1. Implementation Details....91
6.2. Model Repository....92
6.3. Clinical Study Repository....94
6.4. Clinical Decision Support Repository....96
6.6. References....98
7. Results....99
7.1. Basic CRUD workflows....99
7.1.1 Model Repository....99
7.1.1.1 Model....99
7.1.1.2 Parameter....100
7.1.1.3 File....101
7.1.1.4 Property....102
7.1.1.5 Model – Property combination....104
7.1.1.6 Reference....105
7.1.2. Clinical Study Repository....106
7.1.2.1. Clinical Study....106
7.1.2.2. Anonymized Patients....107
7.1.2.3. Stratification....108
7.1.2.4. Patient Partitioning....109
7.1.3.5. Subject....111
7.1.3. Clinical Decision Support Repository....113
7.1.3.1. Clinical Question....113
7.1.3.2. Clinical Question Answer....114
7.2 Model execution management workflows....116
7.2.1 Model Execution and Reports....116
7.2.2 Clinical study support....119
7.3 References....124
8. Conclusions....125
8.1 Discussion....125
8.2 Future expansions....125
8.2.1 Optimization for handling multiple executions....125
8.2.2 Scalable Analysis and Reporting of Model Results....126
8.2.3 Integration of Semantic Infrastructure into the Application Layer....126
8.2.4 From Model-Based Clinical Questions to AI-Driven Hyper-Questions....127
8.3 References....128
Appendix A: Hyperion database table structures....131
Appendix B: List of Abbreviations....138
Appendix C: Publications....140

List of Figures

Figure 1-1. Block diagram of the Oncosimulator's function.....40
Figure 2-1. Conceptual diagram of the proposed architecture and database schema.....51
Figure 2-2. Architecture infrastructure.....51
Figure 2-3. Modeler Workflow.....57
Figure 2-4. Clinician Workflow.....58
Figure 3-1. Schematic diagram of the proposed architecture layout.....63
Figure 4-1: Integration procedure of breast cancer and nephroblastoma Oncosimulators….76
Figure 4-2: Breast cancer Oncosimulator: details of the two simulated bevacizumab monotherapy schemes (dosage, frequency of administration and time points of administration).....77
Figure 4-3: Breast cancer Oncosimulator: tumor volume time evolution for the two simulated bevacizumab schemes.....77
Figure 4-4: Wilms tumor Oncosimulator: tumor evolution graph over time for all three simulated delay cases.....78
Figure 5-1: Functionality diagram of the proposed clinical question module.....85
Figure 5-2: User workflow.....86
Figure 6-1: Hyperion layered architectural diagram.....91
Figure 6-3: Clinical study repository ER diagram.....95
Figure 6-4: Clinical study – model repository coupling ER diagram.....96
Figure 6-5: Clinical decision support repository ER diagram.....97
Figure 7-1: Model list screen.....99
Figure 7-2: Add/Edit model screen.....99
Figure 7-3: Model deletion confirmation pop-up. Similar popups are being produced for all listed entities across Hyperion.....100
Figure 7-4: Parameter list screen with pagination and model-based filtering.....100
Figure 7-5: Add/Edit parameter screen.....101
Figure 7-6: File list screen with pagination and model-based filtering.....102
Figure 7-7: Edit/download file screen.....102
Figure 7-8: Add file screen, with the OS browser open to choose a file to upload.....102
Figure 7-9: Property list screen.....103
Figure 7-10: Add/Edit property screen.....103
Figure 7-11: Model – property combination list screen.....104
Figure 7-12: Model – property combination edit screen.....104
Figure 7-13: Add model-property combinations screen.....105
Figure 7-14: Reference list screen.....105
Figure 7-15: Add/Edit reference screen. 106
Figure 7-16: Clinical Study list screen.....106
Figure 7-17: Add Clinical Study screen.....107
Figure 7-18: Edit Clinical Study screen.....107
Figure 7-19: Add/Edit anonymized patient screen.....108
Figure 7-20: Anonymized patient list screen.....108
Figure 7-21: Stratification list screen.....109
Figure 7-22: Add/Edit stratification screen. 109
Figure 7-23: Anonymized patient – stratification combination list screen.....110
Figure 7-24: Add anonymized patient – stratification combinations screen.....111
Figure 7-25: Edit anonymized patient – stratification combination screen.....111
Figure 7-26: Subject list screen.....112

Figure 7-27: Add subject screen. 112
Figure 7-28: Edit subject screen....113
Figure 7-29: Clinical question list screen....114
Figure 7-30: Add/Edit clinical question screen....114
Figure 7-31: Clinical question answer list screen....115
Figure 7-32: Add clinical question answer screen....115
Figure 7-33: Edit clinical question answer screen....116
Figure 7-34: Model Execution Wizard data input screen for the Wilms Tumor Oncosimulator....117
Figure 7-35: Model Execution list screen....118
Figure 7-36: Model Execution report screen (part 1). The Output file area of the Wilms Tumor Oncosimulator contains a spinning image of the superimposed images of initial and final tumor volumes, as well as a tumor volume evolution graph over time....118
Figure 7-37: Model Execution report screen (part 2)....119
Figure 7-38: Pair plot results (part 1)....122
Figure 7-39: Pair plot results (part 2)....122
Figure 7-40: Pair plot results (part 3)....122
Figure 7-41: Pair plot results (part 4)....123
Figure 7-42: Pair plot results (part 5)....123
Figure 7-43: Scatter plot results for the tumor final growth fraction as per the different vincristine cell kill ratio values for the entire clinical study (no specific stratification or anonymized patient chosen)....124
Figure 7-44: Box plot diagram of tumor volume reduction for all the executions pertaining to a set of patients included in a clinical study....124

List of Tables

Table 4-1: Wilms tumor Oncosimulator: input data for the different delay cases.....77
Table 4-2: Wilms tumor Oncosimulator: result data for the different delay cases.....78
Table 7-1: Initial input values of the Nephroblastoma Oncosimulator.....121

Εκτενής Περίληψη

Η παρούσα διδακτορική διατριβή εμβαθύνει στον τομέα της in silico ογκολογίας και της in silico ιατρικής εστιάζοντας στις τεχνολογίες που απαιτούνται για την ανάπτυξη ολοκληρωμένων Ογκοπροσομοιωτών. Οι τελευταίοι θεωρούνται ως προηγμένα ψηφιακά δίδυμα που συμμετέχουν στην καταπολέμηση του καρκίνου, σχεδιασμένα να βελτιώνουν τον τρόπο με τον οποίο οι κλινικοί γιατροί προσεγγίζουν εξατομικευμένες στρατηγικές θεραπείας. Το βασικό πρόβλημα που αντιμετωπίζει αυτή η επιστήμη πηγάζει από μια θεμελιώδη πρόκληση, εγγενή σε οποιοδήποτε μοντέλο προσομοίωσης: ενώ τα ανεπεξέργαστα δεδομένα εισόδου και εξόδου περιέχουν κλινική γνώση, αυτή συχνά δεν είναι άμεσα προσβάσιμη ή κατανοητή σε ερευνητές ή ιατρούς. Αυτή η έλλειψη προσβασιμότητας εμποδίζει σημαντικά την άμεση κλινική εφαρμογή των μοντέλων στην εξατομικευμένη in silico ιατρική. Προς κάλυψη αυτού του κενού, η παρούσα διατριβή προτείνει και παραθέτει τις λεπτομέρειες υλοποίησης μιας υπολογιστικής υποδομής, η οποία είναι σχεδιασμένη για να αποθηκεύει και να διαχειρίζεται τα διαφορετικά χαρακτηριστικά ενός προσομοιωτικού μοντέλου (παράμετροι, εκτελέσιμα αρχεία, βασικές περιγραφικές πληροφορίες, κτλ.), συμπεριλαμβανομένων και των δεδομένων των εκτελέσεων τους.

Τα αρχικά κεφάλαια αυτής της εργασίας παρουσιάζουν τις θεμελιώδεις αρχές που διέπουν το σχεδιασμό, την υλοποίηση και τις λειτουργικότητες μιας τέτοιας υποδομής. Κατόπιν παρατίθενται προτεινόμενες επεκτάσεις που αποσκοπούν στην ενίσχυση των δυνατοτήτων του συστήματος, και συγκεκριμένα στο χειρισμό σημασιολογικών δεδομένων, τον ορισμό κλινικών ερωτημάτων και σχετικών απαντήσεων βάσει των χαρακτηριστικών των μοντέλων, καθώς και τη διαχείριση δεδομένων πολλαπλών εκτελέσεων στα πλαίσια μιας κλινικής δοκιμής. Η πρακτική χρησιμότητα όλου του εγχειρήματος, συμπεριλαμβανομένων των Ογκοπροσομοιωτών παρουσιάζεται μέσα από τα αποτελέσματα ευρωπαϊκών ερευνητικών έργων. Αυτά τα παραδείγματα υποδεικνύουν πώς η προτεινόμενη υποδομή μπορεί να χρησιμεύσει ως βασικό συστατικό της μετάβασης αυτών των υπολογιστικών εργαλείων από το ερευνητικό εργαστήριο στην καθημερινή κλινική πράξη, όπου μπορούν να λειτουργήσουν ως συστήματα υποστήριξης κλινικών αποφάσεων. Η διατριβή παρουσιάζει επίσης μια πρωτότυπη υλοποίηση αυτής της υποδομής, παρέχοντας μια λεπτομερή περιγραφή των τεχνικών σχεδιασμού και των εργαλείων ανάπτυξης που χρησιμοποιήθηκαν (γλώσσες προγραμματισμού, κτλ.), της ανάλυσής της σε διακριτά, διαχειρίσιμα στοιχεία και μια ολοκληρωμένη έκθεση διαφόρων περιπτώσεων χρήσης μαζί με τα αποτελέσματά τους. Η διατριβή ολοκληρώνεται με μια συνοπτική περίληψη των ευρημάτων και σκιαγραφεί μελλοντικές επεκτάσεις και κατευθύνσεις για περαιτέρω έρευνα και ανάπτυξη σε αυτόν τον ζωτικό για την έννοια της εξατομικευμένης ιατρικής τομέα.

In Silico Ογκολογία: Ορισμός και στόχοι

Η σύγχρονη ογκολογία παρουσιάζει σήμερα στροφή προς την εξατομικευμένη ιατρική. Αυτή η μετάβαση προς μια νέα προσέγγιση καθοδηγείται από μια συνεχώς βαθύτερη κατανόηση των πολύπλοκων μηχανισμών στις οποίες βασίζεται ο καρκίνος. Μέσα σε αυτό το δυναμικό τοπίο, η in silico ογκολογία έχει αναδειχθεί ως ένα πολλά

υποσχόμενο επιστημονικό πεδίο. Αξιοποιεί υπολογιστική μοντελοποίηση και προηγμένες τεχνικές προσομοίωσης για να συμπεριλάβει την βιολογικά πολυεπίπεδη πολυπλοκότητα της εξέλιξης του καρκίνου και της ανταπόκρισής του στη θεραπεία, επιτρέποντας έτσι την ανάπτυξη και εφαρμογή εξατομικευμένων στρατηγικών αντιμετώπισης. Ο πρωταρχικός στόχος αυτής της προσέγγισης είναι πολύπλευρος: να ενισχύσει σημαντικά την αποτελεσματικότητα των θεραπειών για τον καρκίνο, να ελαχιστοποιήσει τις παρενέργειες που βιώνουν οι ασθενείς και να επιταχύνει τον ρυθμό ανάπτυξης νέων φαρμάκων. Αυτό επιτυγχάνεται με τη δημιουργία εικονικών αναπαραστάσεων ή ψηφιακών διδύμων των όγκων και τη προσομοίωση των πολύπλοκων αλληλεπιδράσεών τους με διάφορες θεραπευτικές παρεμβάσεις. Στην ουσία, η in silico ογκολογία επεκτείνει τις αρχές της μαθηματικής μοντελοποίησης των βιολογικών συστημάτων για να εξηγήσει και να προβλέψει δυναμικές συμπεριφορές του καρκίνου. Αυτό γίνεται εστιάζοντας σε κλινικά καθοδηγούμενα, πολυκλιμακωτά μοντέλα προσομοίωσης κακοηθών όγκων, λαμβάνοντας υπόψιν ότι ο καρκίνος είναι μια ασθένεια που εκδηλώνεται σε πολλαπλές βιολογικές κλίμακες, από μοριακή και κυτταρική έως επίπεδο οργάνων.

Ο απώτερος στόχος αυτής της επιστήμης είναι να χρησιμοποιήσει αυτά τα εξαιρετικά λεπτομερή μοντέλα, μετά από αυστηρή κλινική προσαρμογή και επικύρωση ως ασθενοκεντρικά συστήματα υποστήριξης αποφάσεων και εργαλεία σχεδιασμού θεραπείας. Αυτό περιλαμβάνει την δυνατότητα προσομοίωσης ογκολογικών κλινικών δοκιμών, μια διαδικασία που διαφορετικά θα ήταν δαπανηρή, ηθικά περίπλοκη και χρονοβόρα εάν διεξαγόταν αποκλειστικά σε παραδοσιακά εργαστήρια ή κλινικά περιβάλλοντα. Η αποτελεσματική εκτέλεση αυτών των πολύπλοκων προσομοιώσεων συχνά διευκολύνεται από προηγμένες κατανεμημένες υπολογιστικές υποδομές, όπως η υπολογιστική πλέγματος (grid computing).

Η γένεση της in silico ογκολογίας μπορεί να αναχθεί στην πρωτοποριακή έρευνα που διεξήχθη στο Εθνικό Μετσόβιο Πολυτεχνείο (ΕΜΠ). Ο πρωταρχικός όρος "in silico radiation oncology" εισήχθη για πρώτη φορά το 2002, θέτοντας τις θεμελιώδεις εννοιολογικές και μεθοδολογικές βάσεις για αυτό που αργότερα θα εξελισσόταν στον ευρύτερο τομέα της *in silico* ιατρικής. Ο Ογκοπροσομοιωτής, μια υπολογιστική πλατφόρμα που αναπτύχθηκε από την Ομάδα In Silico Ογκολογίας και In Silico Ιατρικής του Ερευνητικού Πανεπιστημιακού Ινστιτούτου Συστημάτων Επικοινωνιών και Υπολογιστών (ΕΠΙΣΕΥ), έγινε γρήγορα ένα κεντρικό και απαραίτητο συστατικό πολλών μεγάλων ολοκληρωμένων ερευνητικών έργων που χρηματοδοτήθηκαν από την Ευρωπαϊκή Ένωση. Αυτά περιελάμβαναν πρωτοβουλίες με μεγάλο αντίκτυπο όπως το ACGT (Advancing Clinico-Genomic Trials on Cancer), το ContraCancrum (Clinically Oriented Translational Cancer Multilevel Modelling), το CHIC (Computational Horizons In Cancer), το MyHealthAvatar (A Demonstration of 4D Digital Avatar Infrastructure for Access of Complete Patient Information) και το P-Medicine (From data sharing and integration via VPH models to personalized medicine). Κάθε ένα από αυτά τα έργα είχε έναν κοινό στόχο: να αναπτύξει, να βελτιώσει και να επικυρώσει κλινικά προηγμένες πλατφόρμες πολυκλιμακωτής προσομοίωσης που έχουν σχεδιαστεί ρητά για να υποστηρίζουν εξατομικευμένο σχεδιασμό θεραπείας για ασθενείς με καρκίνο. Μεταγενέστερα έργα, όπως

το DR THERAPAT (Digital Radiation Therapy Patient και το BOUNCE (Predicting Effective Adaptation to Breast Cancer to Help Women to BOUNCE back), επέκτειναν περαιτέρω το πεδίο εφαρμογής των in silico ιατρικών προσεγγίσεων, αναδεικνύοντας τις δυνατότητες αυτού του τομέα.

Πέρα από τις ερευνητικές συνεισφορές του, ο τομέας της in silico ογκολογίας έχει πετύχει επίσης αξιοσημείωτα ακαδημαϊκά επιτεύγματα. Η Ελλάδα, μέσω της πρωτοβουλίας του ΕΜΠ, είναι η πρώτη χώρα παγκοσμίως που φιλοξενεί ένα πανεπιστημιακού επιπέδου μάθημα ειδικά αφιερωμένο σε αυτόν τον αναδυόμενο τομέα. Το μεταπτυχιακό μάθημα, με τίτλο "Multiscale Cancer Modelling and In Silico Medicine", εισήχθη στο ΕΜΠ το 2014 και συνεχίζει να διδάσκεται, εκπαιδεύοντας την επόμενη γενιά ερευνητών και κλινικών ιατρών σε αυτόν τον διεπιστημονικό τομέα. Αυτή η ακαδημαϊκή πρωτοβουλία σύντομα κέρδισε διεθνή αναγνώριση. Το Virtual Physiological Human Institute (VPHi), ένας διεθνής οργανισμός αφιερωμένος στην προώθηση της in silico ιατρικής, ανέδειξε ρητά την υγειονομική περίθαλψη που βασίζεται στην προσομοίωση ως στρατηγική κατεύθυνση για μελλοντικές ιατρικές εξελίξεις. Ταυτόχρονα, η Avicenna Alliance, ένας ευρωπαϊκός βιομηχανικός φορέας, ενέκρινε επίσημα την in silico ιατρική, αναγνώρισε την Ελλάδα και το ΕΜΠ ως την κοιτίδα της και συμμετείχε ενεργά στην ανάπτυξη ισχυρών ρυθμιστικών πλαισίων για τη διευκόλυνση της ευρείας υιοθέτησης και αποδοχής in silico κλινικών δοκιμών. Αυτές οι εξελίξεις υπογραμμίζουν την αυξανόμενη δυναμική και τον κρίσιμο ρόλο που η in silico ογκολογία είναι έτοιμη να αποτελέσει στο μέλλον της εξατομικευμένης υγειονομικής περίθαλψης.

Ο Ογκοπροσομοιωτής: Ένα ψηφιακό δίδυμο για εξατομικευμένη φροντίδα του καρκίνου

Ο Ογκοπροσομοιωτής αποτελεί τον ακρογωνιαίο λίθο στον αναπτυσσόμενο τομέα της in silico ογκολογίας. Είναι σχεδιασμένος για να προσομοιώνει τις πολύπλοκες αποκρίσεις τόσο ενός καρκινικού όγκου όσο και των υγιών φυσιολογικών ιστών που τον περιβάλλουν σε ένα ευρύ φάσμα θεραπευτικών στρατηγικών. Αυτό που ξεχωρίζει τον Ογκοπροσομοιωτή είναι η ικανότητα να προσαρμόζει αυτές τις προσομοιώσεις με ακρίβεια σε κάθε μεμονωμένο ασθενή, αξιοποιώντας το κλινικό ιστορικό τους, τα αρχικά απεικονιστικά δεδομένα (αξονικές και μαγνητικές τομογραφίες, κτλ.) καθώς και τα μοριακά και ιστοπαθολογικά ευρήματα που έχουν προκύψει από βιοψίες. Η πρωταρχική αποστολή του είναι να επιτύχει τη βέλτιστη θεραπεία του καρκίνου για κάθε ασθενή, όχι μέσω δοκιμών σε έναν ζωντανό οργανισμό, αλλά εφαρμόζοντας μέσω πειραμάτων σε υπολογιστή (εξού και ο χαρακτηρισμός in silico) ενός πλήθους πιθανών θεραπευτικών προσεγγίσεων. Πέρα από τη χρησιμότητά του στην υποστήριξη κλινικών αποφάσεων, ο Ογκοπροσομοιωτής εξυπηρετεί επίσης έναν ευρύτερο σκοπό: στοχεύει να ενισχύσει σημαντικά την επιστημονική κατανόηση της βιολογίας του καρκίνου, να διευκολύνει τον σχεδιασμό κλινικογονιδιωματικών δοκιμών και να παρέχει ένα εργαλείο για ιατρική εκπαίδευση. Βασιζόμενος στο παράδειγμα του ψηφιακού διδύμου, ο Ογκοπροσομοιωτής δημιουργεί ένα δυναμικό, εικονικό αντίγραφο του όγκου ενός ασθενούς, επιτρέποντας την

ακριβή προσομοίωση της εξέλιξης του όγκου και την προβλεπόμενη ανταπόκρισή του σε διάφορες θεραπείες για καθορισμένες χρονικές περιόδους.

Η λειτουργία του Ογκοπροσομοιωτή είναι δομημένη γύρω από μια διαδικασία επτά βημάτων:

1. **Συλλογή δεδομένων:** Αυτό το αρχικό βήμα αφορά στη λήψη δεδομένων ενός ασθενούς που περιλαμβάνουν ένα ευρύ φάσμα πληροφοριών, συμπεριλαμβανομένων λεπτομερών κλινικών αρχείων, διαφόρων ιατρικών απεικονιστικών σαρώσεων (όπως CT, MRI, PET), ιστοπαθολογικών ευρημάτων από βιοψίες ιστών και μοριακών δεδομένων (π.χ. γονιδιωματικά, πρωτεομικά, μεταγραφωμικά προφίλ). Η ποιότητα και η πληρότητα αυτών των δεδομένων επηρεάζουν άμεσα την ακρίβεια των προσομοιώσεων.
2. **Προεπεξεργασία δεδομένων:** Μόλις αποκτηθούν τα δεδομένα του ασθενούς υποβάλλονται σε ένα στάδιο προεπεξεργασίας. Κατά τη διάρκεια αυτής της φάσης, τα ετερογενή δεδομένα μετατρέπονται σε μια δομημένη μορφή που είναι άμεσα συμβατή με τις απαιτήσεις του Ογκοπροσμοιωτή. Αυτό συχνά περιλαμβάνει ανωνυμοποίηση, αποχαρακτηρισμό και έλεγχο σφαλμάτων για τη διασφάλιση της ακεραιότητας των δεδομένων και του απορρήτου των ασθενών.
3. **Ορισμός θεραπευτικού σχήματος:** Σε αυτό το βήμα, ένας κλινικός ιατρός, με βάση την εμπειρία του και την αρχική διάγνωση του ασθενούς, καθορίζει ένα σύνολο υποψήφιων θεραπευτικών σχημάτων. Αυτά μπορεί να περιλαμβάνουν διάφορα σχέδια ακτινοθεραπείας, σχήματα χημειοθεραπείας, πρωτόκολλα ανοσοθεραπείας ή συνδυασμούς αυτών, τα οποία στη συνέχεια θα δοκιμαστούν στο εικονικό περιβάλλον.
4. **Εκτέλεση προσομοίωσης:** Τα καθορισμένα σενάρια θεραπείας εκτελούνται σε ισχυρούς, κατανεμημένους υπολογιστικούς πόρους, συνήθως χρησιμοποιώντας αρχιτεκτονικές πλέγματος (grid) ή συμπλέγματος (cluster). Αυτή η ικανότητα επεξεργασίας είναι απαραίτητη, καθώς επιτρέπει την ταυτόχρονη και ταχεία αξιολόγηση πολλών θεραπευτικών επιλογών και ενός ευρέος φάσματος συνδυασμών παραμέτρων του όγκου. Έτσι μειώνεται σημαντικά ο υπολογιστικός χρόνο, καθιστώντας τις προσομοιώσεις κλινικά αξιοποιήσιμες.
5. **Οπτικοποίηση πρόβλεψης:** Μετά την ολοκλήρωση των προσομοιώσεων, η προβλεπόμενη απόκριση του όγκου και οι πιθανές τοξικολογικές παρενέργειες για όλα τα προσομοιωμένα σενάρια απεικονίζονται λεπτομερώς χρησιμοποιώντας αριθμητικά διαγράμματα, και τρισδιάστατες στατικές και δυναμικές απεικονίσεις της εικόνας του όγκου καθ'όλη τη διάρκεια της προσομοίωσης. Ο στόχος είναι να παρουσιαστούν πολύπλοκα αποτελέσματα προσομοίωσης με σαφή, εφαρμόσιμο και κλινικά κατανοητό τρόπο.
6. **Κλινική αξιολόγηση:** Εφοδιασμένοι με τις οπτικοποιημένες προβλέψεις, οι κλινικοί γιατροί εισέρχονται στη φάση της αξιολόγησης των αποτελεσμάτων. Συνδυάζουν τις εκτεταμένες ιατρικές γνώσεις, την κλινική εμπειρία και τις εξατομικευμένες πληροφορίες του ασθενούς με τις in silico προβλέψεις για να λάβουν τεκμηριωμένη απόφαση σχετικά με το βέλτιστο σχέδιο θεραπείας. Αυτό το βήμα υπογραμμίζει το

ρόλο του Ογκοπροσομοιωτή ως εργαλείου υποστήριξης αποφάσεων, αυξάνοντας την ανθρώπινη τεχνογνωσία αντί να την αντικαθιστά.

7. **Χορήγηση θεραπείας και ανατροφοδότηση:** Το επιλεγμένο, βελτιστοποιημένο σχέδιο θεραπείας χορηγείται στη συνέχεια στον ασθενή. Κατόπιν, συλλέγονται συστηματικά δεδομένα που συγκρίνουν τις προβλέψεις του Ογκοπροσομοιωτή με τα πραγματικά κλινικά αποτελέσματα και τις αποκρίσεις των ασθενών. Αυτός ο συνεχής βρόχος ανατροφοδότησης είναι ζωτικής σημασίας για την επαναλαμβανόμενη βελτίωση της ακρίβειας πρόβλεψης και της συνολικής απόδοσης του Ογκοπροσομοιωτή, διασφαλίζοντας την κλινική χρησιμότητά του.

Ο Ογκοπροσομοιωτής έχει σαν βάση του την "top-down" πολυκλιμακωτή στρατηγική προσομοίωσης, μια πρωτοποριακή προσέγγιση που αναπτύχθηκε και τελειοποιήθηκε στο ΕΜΠ. Αυτή η στρατηγική αναγνωρίζει ότι ο καρκίνος είναι μια πολυκλιμακωτή ασθένεια, που απαιτεί μοντελοποίηση από το μοριακό και κυτταρικό επίπεδο μέχρι το επίπεδο ιστών και οργάνων. Οι θεμελιώδεις διαδικασίες αυτής της μοντελοποίησης περιλαμβάνουν την περίπλοκη διαταραχή (perturbation) των ραδιοβιολογικών ή φαρμακοδυναμικών παραμέτρων κυτταρικής θανάτωσης, οι οποίες προσαρμόζονται δυναμικά με βάση τα μοριακά δεδομένα του μεμονωμένου ασθενή. Η προσομοίωση χρησιμοποιεί ένα σύστημα για την ποσοτικοποίηση κυτταρικών συστάδων μέσα σε ένα διακριτικό πλέγμα που καλύπτει με ακρίβεια την ανατομική περιοχή ενδιαφέροντος. Μέσα σε αυτό το πλέγμα, οι διάρκειες φάσης του κυτταρικού κύκλου και τα δεδομένα μεταβολισμού χρησιμοποιούνται για τον καθορισμό την πορείας που θα ακολουθηθεί από τα κύτταρα κάθε μονάδας αυτού του πλέγματος. Στη συνέχεια χρησιμοποιούνται αλγόριθμοι για την προσομοίωση μακροσκοπικών μηχανισμών, συμπεριλαμβανομένης της δυναμικής επέκτασης και συρρίκνωσης του όγκου, της επιβολής μηχανικών οριακών συνθηκών στους ιστούς και των ακριβών επιπτώσεων διαφόρων αντικαρκινικών φαρμάκων και ακτινοβολίας τόσο σε καρκινικά όσο και σε υγιή κύτταρα.

Μαθηματικά, ο Ογκοπροσομοιωτής ενσωματώνει μια σειρά υπολογιστικών τεχνικών. Αυτές περιλαμβάνουν, μεταξύ άλλων, μη ντετερμινιστικά αυτόματα πεπερασμένων καταστάσεων για τη μοντελοποίηση σύνθετων κυτταρικών συμπεριφορών, τεχνικές Monte Carlo για προσομοίωση στοχαστικών διεργασιών όπως η εναπόθεση ακτινοβολίας και ο κυτταρικός θάνατος και διαφορικές εξισώσεις για την περιγραφή συνεχών αλλαγών στις βιολογικές παραμέτρους με την πάροδο του χρόνου. Από τεχνολογική άποψη, ο Ογκοπροσομοιωτής αξιοποιεί πολλές τεχνολογίες αιχμής που συμπεριλαμβάνουν δυναμικά και πολυδιάστατα εργαλεία οπτικοποίησης για την εισαγωγή ιατρικών δεδομένων και προβλέψεις προσομοίωσης, προηγμένες τεχνικές επεξεργασίας ιατρικής εικόνας για την εξαγωγή κρίσιμων ανατομικών και φυσιολογικών πληροφοριών, μηχανισμούς επιτάχυνσης εκτέλεσης (όπως αρχιτεκτονικές πλέγματος για κατανεμημένους υπολογιστές) και ισχυρά συστήματα για την ασφαλή και νομικά συμβατή μεταφορά και αποθήκευση εξαιρετικά ευαίσθητων βιοϊατρικών δεδομένων (συμπεριλαμβανομένων εξελιγμένων τεχνικών

Τεχνολογικές προκλήσεις της μετάβασης προς την εξατομικευμένη ιατρική

Παρά τις πρωτοποριακές εξελίξεις στην in silico ογκολογία και την πολλά υποσχόμενη συμβολή της στην εξατομικευμένη ιατρική, αρκετά σημαντικά τεχνολογικά εμπόδια εμποδίζουν σήμερα την ευρεία υιοθέτησή της και την πλήρη κλινική υλοποίησή της. Μια σημαντική και πρωταρχική πρόκληση έγκειται στην έλλειψη μιας ισχυρής, κλιμακούμενης και τυποποιημένης υποδομής ικανής να φιλοξενεί αποτελεσματικά υπολογιστικά μοντέλα, να εκτελεί πολύπλοκες προσομοιώσεις εντός κλινικά αποδεκτών χρονικών πλαισίων και να παράγει ολοκληρωμένες, αξιόπιστες και κλινικά αξιοποιήσιμες αναφορές από τα παραγόμενα αποτελέσματα. Η διαχείριση του τεράστιου όγκου και της εγγενούς ετερογένειας των δεδομένων που αφορούν τον ασθενή, η οποία κυμαίνεται από γονιδιωματικές αλληλουχίες έως λεπτομερείς σαρώσεις απεικόνισης και ηλεκτρονικά αρχεία υγείας, παρουσιάζει μια τεράστια υλικοτεχνική και υπολογιστική πολυπλοκότητα. Επιπλέον, ενώ οι υπολογιστικοί πόροι είναι αναμφίβολα απαραίτητοι για την εκτέλεση αυτών των περίπλοκων προσομοιώσεων, ιδίως όταν οι ταχείς χρόνοι διεκπεραίωσης είναι κρίσιμοι για την περίθαλψη των ασθενών, η πρόσβαση σε τέτοια περιβάλλοντα υπολογιστικής υψηλών επιδόσεων (HPC) εξακολουθεί να αποτελεί σημαντικό περιορισμό για πολλά κλινικά περιβάλλοντα. Επιπλέον, η εγγενής ποικιλομορφία των μοντέλων in silico, καθένα από τα οποία αντιπροσωπεύει διαφορετικές βιολογικές κλίμακες, φυσιολογικές διεργασίες ή θεραπευτικές μεθόδους, δημιουργεί σημαντική δυσκολία όσον αφορά την απρόσκοπτη ολοκλήρωση και διαλειτουργικότητά τους. Τέλος, η πρόκληση της μετατροπής των ακατέργαστων αποτελεσμάτων προσομοίωσης σε τυποποιημένες, ερμηνεύσιμες και κλινικά εφαρμόσιμες αναφορές για ιατρούς παραμένει ένας κρίσιμος τομέας που απαιτεί ουσιαστική περαιτέρω ανάπτυξη και βελτίωση.

Η αντιμετώπιση αυτών των κεντρικών εμποδίων απαιτεί μια ολοκληρωμένη και προοδευτική κατανόηση των θεωρητικών απαιτήσεων σε διάφορους κρίσιμους τομείς:

- **Διαχείριση δεδομένων:** Απαιτείται η καθιέρωση αυστηρών πρωτοκόλλων τυποποίησης για τις μορφές δεδομένων, την ονοματολογία και τις σχολιασμούς για τη διασφάλιση της σημασιολογικής διαλειτουργικότητας μεταξύ διαφορετικών συστημάτων και ιδρυμάτων. Επιπλέον, η ανάπτυξη ασφαλών, ανθεκτικών μηχανισμών αποθήκευσης και ισχυρών στρατηγικών διαχείρισης για συνεχώς αυξανόμενα σύνολα δεδομένων είναι θεμελιώδης, ειδικά παράλληλα με την διατήρηση του απορρήτου των ασθενών και την τήρηση αυστηρών κανονισμών προστασίας δεδομένων (π.χ., GDPR, HIPAA). Η καθιέρωση ολοκληρωμένων πλαισίων κοινής χρήσης δεδομένων που συμμορφώνονται πλήρως με τις αρχές FAIR (Εύρεση, Προσβασιμότητα, Διαλειτουργικότητα, Επαναχρησιμοποίηση) είναι απαραίτητη για την ενίσχυση των συνεργατικών ερευνητικών πρωτοβουλιών και την επιτάχυνση της επιστημονικής ανακάλυψης.
- **Ενσωμάτωση μοντέλων:** Απαιτούνται αποτελεσματικοί μηχανισμοί για την απρόσκοπτη ενσωμάτωση πολυκλιμακωτών μοντέλων, τα οποία εγγενώς αντιπροσωπεύουν διαφορετικές επίπεδα βιοπολυπλοκότητας (μοριακό, κυτταρικό, ιστών, κλπ.). Κατ' επέκταση χρειάζεται η ανάπτυξη ανοιχτών πλατφορμών που υποστηρίζουν ενεργά την αποτελεσματική κοινή χρήση και την πρακτική

επαναχρησιμοποίηση μοντέλων in silico για την επίτευξη πραγματικής σημασιολογικής διαλειτουργικότητας μεταξύ διαφορετικών προσεγγίσεων μοντελοποίησης, μαθηματικών φορμαλισμών και εργαλείων προσομοίωσης

- **Υπολογιστικοί πόροι:** Οι in silico προσομοιώσεις απαιτούν ισχυρή υποδομή υπολογιστικής υψηλής απόδοσης (HPC). Αυτή η υποδομή πρέπει να παρέχει επεκτάσιμους πόρους, εξειδικευμένο λογισμικό για την υποδοχή και εκτέλεση μοντέλων, καθώς και προηγμένα εργαλεία ανάλυσης. Η βελτιστοποίηση αλγορίθμων και τεχνικών παραλληλοποίησης είναι απαραίτητη για την επίτευξη ταχύτητας και την εφαρμογή των αποτελεσμάτων σε χρονικά πλαίσια που έχουν νόημα για την καθημερινή κλινική πρακτική.
- **Ηθικά ζητήματα:** Κρίσιμης σημασίας είναι η προστασία της ιδιωτικότητας και της ασφάλειας των ευαίσθητων δεδομένων υγείας, καθώς και ο περιορισμός των προκαταλήψεων σε μοντέλα Τεχνητής Νοημοσύνης. Η διαφάνεια και η εξηγησιμότητα ενισχύουν την εμπιστοσύνη των χρηστών, ενώ τα ηθικά πλαίσια πρέπει να διασφαλίζουν την ισορροπία μεταξύ επιστημονικής προόδου και ατομικών δικαιωμάτων. Τέλος, η λογοδοσία και η πλήρως ενημερωμένη συγκατάθεση των ασθενών αποτελούν ηθικές επιταγές.

Η τρέχουσα κατάσταση στην εξατομικευμένη ογκολογία

Η σύγχρονη εξατομικευμένη ογκολογία χαρακτηρίζεται από ραγδαία ενσωμάτωση προηγμένων τεχνολογιών, με τα Συστήματα Υποστήριξης Κλινικών Αποφάσεων (CDSS), όπως το IBM Watson for Oncology και το CSCO AI, να αξιοποιούν μεγάλα σύνολα δεδομένων για την παροχή προσωποποιημένων θεραπευτικών συστάσεων. Παράλληλα, συστήματα όπως το PREDICT εστιάζουν σε πιο εξειδικευμένες ανάγκες, ενώ νεότερα CDSS που βασίζονται σε μεγάλα γλωσσικά μοντέλα και μηχανική μάθηση ενισχύουν τη δυνατότητα πρόβλεψης και εξατομίκευσης.

Επιπλέον, η Τεχνητή Νοημοσύνη χρησιμοποιείται σε διαγνωστικά απεικόνισης, στη δημιουργία λεπτομερών προφίλ ασθενών μέσω γονιδιωματικών, απεικονιστικών και κλινικών δεδομένων, καθώς και στη λειτουργική ιατρική ακριβείας, η οποία συνδυάζει γενετική ανάλυση και ex vivo δοκιμές φαρμάκων. Άλλες σημαντικές τεχνολογίες περιλαμβάνουν την τηλεϊατρική και την απομακρυσμένη παρακολούθηση, τη ρομποτική χειρουργική, τα προηγμένα συστήματα ηλεκτρονικών φακέλων υγείας (ΗΦΥ) με ενσωματωμένα εργαλεία CDSS, καθώς και εξειδικευμένες εφαρμογές κινητών για ογκολόγους και ασθενείς. Η Τεχνητή Νοημοσύνη ενισχύει επίσης την ανακάλυψη και επαναχρησιμοποίηση φαρμάκων, ενώ οι υγρές βιοψίες προσφέρουν μη επεμβατικές λύσεις για διάγνωση και παρακολούθηση καρκίνου. Τέλος, τεχνολογίες γονιδιακής επεξεργασίας όπως το CRISPR ανοίγουν νέους ορίζοντες, παρότι βρίσκονται ακόμη σε ερευνητικό στάδιο.

Παρά τις σημαντικές προόδους, η πλήρης κλινική αξιοποίηση αυτών των εργαλείων εξαρτάται από την προσαρμογή τους στις υπάρχουσες ροές εργασίας, τη συνέπεια με τις κλινικές οδηγίες, την εξασφάλιση διαφάνειας και αξιοπιστίας, αλλά και την υιοθέτηση

πολυπαραγοντικών προσεγγίσεων, που ενισχύουν την ακρίβεια και την εξατομίκευση στη φροντίδα του καρκίνου.

Εννοιολογική επισκόπηση συστήματος της προτεινόμενης υποδομής

Η βασική ιδέα της προτεινόμενης υποδομής ενσωματώνει σχεσιακές βάσεις δεδομένων, μια διαδικτυακή εφαρμογή και ένα σύστημα αποθήκευσης αρχείων Η διαδικτυακή εφαρμογή λειτουργεί ως "κέλυφος" γύρω από τη βάση δεδομένων, επιβάλλοντας τις αρχές ACID (Ατομικότητα, Συνέπεια, Απομόνωση, Ανθεκτικότητα) και διαχειριζόμενη όλες τις επικοινωνίες και την επικύρωση δεδομένων εισόδου. Ένα δεύτερο επίπεδο είναι αφιερωμένο στην ασφάλεια του συστήματος. Η αρχιτεκτονική δίνει έμφαση στην τυποποίηση των αποθηκευμένων περιγραφικών πληροφοριών χρησιμοποιώντας ένα κατάλληλο σχήμα σχεσιακής βάσης δεδομένων. Έχει σχεδιαστεί για να χειρίζεται διάφορους τύπους δεδομένων, συμπεριλαμβανομένων των παραμέτρων του μοντέλου (είσοδος και έξοδος) και των σχετικών αρχείων (υλοποιήσεις, τεκμηρίωση, συμπληρωματικά σενάρια. Τα δεδομένα εκτέλεσης, όπως λίστες τιμών παραμέτρων ανά ασθενή και εξειδικευμένες εξόδους προσομοίωσης όπως γραφήματα και αναφορές PDF, αποθηκεύονται συνδυαστικά σε ένα τμήμα της βάσης δεδομένων και ενός συστήματος αρχείων για εύκολη ανάκτηση.

Το σύστημα παρέχει γραφικές διεπαφές χρήστη (GUI) που προσφέρουν μια σειρά λειτουργιών στους τελικούς χρήστες. Η κύρια λειτουργία περιλαμβάνει την επεξεργασία του περιεχομένου της βάσης δεδομένων, όπου δημιουργούνται κατάλληλες προβολές ανά πίνακα, επιτρέποντας στους χρήστες να εκτελούν όλες τις λειτουργίες Δημιουργίας, Ανάγνωσης, Ενημέρωσης και Διαγραφής (CRUD) σε αποθηκευμένα αντικείμενα. Τα δεδομένα εισόδου επικυρώνονται σε σχέση με τις υπάρχουσες πληροφορίες της βάσης δεδομένων ή τους ορισμούς μοντέλων (π.χ. αποτροπή εισαγωγής τιμών βιολογικών παραμέτρων εκτός κλινικά ωφέλιμου εύρους) πριν από την αποθήκευσή τους.

Για τις εκτελέσεις μοντέλων, στους χρήστες παρέχεται μια φόρμα πολλαπλών βημάτων (wizard) που παρουσιάζει βασικές πληροφορίες μοντέλου, παραθέτει παραμέτρους εισόδου και ζητά της συμπλήρωση της για την έναρξη της εκτέλεσης. Τα πεδία εισόδου μπορούν να δημιουργηθούν δυναμικά από τον πίνακα παραμέτρων, διασφαλίζοντας ότι ενημερώνονται πάντα με τις πιο πρόσφατες αλλαγές στο μοντέλο. Μια δεύτερη σχετική φόρμα παρουσιάζει τα αποτελέσματα εκτέλεσης και μπορεί να δημιουργήσει εκτυπώσιμες αναφορές σε μορφή PDF. Τέλος, τα εργαλεία UI επιτρέπουν τη βασική ανάλυση των αποθηκευμένων αποτελεσμάτων εκτέλεσης για οποιονδήποτε αριθμό εκτελέσεων μοντέλων. Ένα αρχικό παράδειγμα υλοποίησης παρουσιάζεται στο πλαίσιο του έργου MyHealthAvatar, συμπεριλαμβανομένων των Ογκοπροσομοιωτών για το νεφροβλάστωμα και τον καρκίνο του μαστού

Δημιουργία μεταδεδομένων για μοντέλα in silico*:* Μια θεωρητική προσέγγιση

Υπάρχει μια κρίσιμη ανάγκη για προηγμένες στρατηγικές διαχείρισης δεδομένων για υπολογιστικά μοντέλα στην in silico ογκολογία, ιδιαίτερα με την άνοδο των υπερμοντέλων που ενσωματώνουν ποικίλες βιολογικές κλίμακες και πτυχές του καρκίνου.

Προτείνεται μια θεωρητική υποδομή που αξιοποιεί τεχνολογίες σημασιολογικού ιστού για την παραγωγή και διαχείριση μεταδεδομένων, προσθέτοντας δυνατότητες στα υπάρχοντα δεδομένα για βελτιωμένη μηχανική κατανόηση και διαλειτουργικότητα. Αυτή η αρθρωτή υποδομή έχει σχεδιαστεί για να λειτουργεί με διάφορα εργαλεία λογισμικού, προσφέροντας ευελιξία σε ερευνητικά περιβάλλοντα. Προβλέπεται να συνδέεται με ένα ευρύτερο οικοσύστημα μεταδεδομένων για την υποστήριξη της δημοσίευσης περιεχομένου, της προηγμένης αναζήτησης και της συμμόρφωσης με τα νομικά και ηθικά πλαίσια για τα δεδομένα υγειονομικής περίθαλψης. Ο σχεδιασμός ενσωματώνει δικαιώματα πρόσβασης για την εξυπηρέτηση διακριτών ενδιαφερομένων, με τους δημιουργούς μοντέλων να έχουν προνόμια αναβάθμισης και συντήρησης, ενώ οι τελικοί χρήστες επωφελούνται από φιλικές προς το χρήστη διεπαφές για την εκτέλεση μοντέλων και την ανάκτηση αποτελεσμάτων. Το συνολικό σύστημα περιλαμβάνει επτά κύριες ενότητες: το Αρχικό Σημείο Πρόσβασης, τη μονάδα Εξαγωγής-Μετασχηματισμού-Φόρτωσης (ETL), δύο αποθετήρια (δεδομένα RDF και βάση γνώσεων) και τρεις εφαρμογές front-end (διαχείριση σχολίων, διαχείριση ερωτημάτων, διαχείριση βάσης γνώσεων).

Διαχείριση κλινικών ερωτήσεων και απαντήσεων

Επεκτείνοντας τη βασική υποδομή αποθήκευσης μοντέλων και εκτελέσεων, παρουσιάζονται οι βασικές αρχές για τον χειρισμό κλινικών ερωτήσεων και των αντίστοιχων απαντήσεών τους. Μέσα από αυτές μπορεί να δημιουργηθεί ένα νέο τμήμα της προτεινόμενης λύσης, του οποίου ο κύριος στόχος είναι να εμπλουτίσει τα αποτελέσματα εκτέλεσης και να τα μεταφράσει σε πολύτιμες κλινικές γνώσεις που μπορούν να εφαρμοστούν άμεσα σε εξατομικευμένες ιατρικές αποφάσεις.

Αυτό επιτυγχάνεται μέσω της συσχέτισης των τιμών (μεμονωμένων ή πεδίων) που μπορεί να λάβει η κάθε παράμετρος ενός μοντέλου με την κατάλληλη περιγραφική πληροφορία ώστε να δημιουργήσει ένα ζεύγος ερώτησης – απάντησης. Αυτά τα ζεύγη κατόπιν θα εμφανίζονται στις αναφορές εκτέλεσης ενός μοντέλου, διευκολύνοντας την εξαγωγή ποιοτικών συμπερασμάτων για την πορεία εξέλιξης ενός θεραπευτικού σχήματος, πέραν της παροχής των ίδιων των αριθμητικών δεδομένων της εκτέλεσης.

Το σχετικό σύστημα διέπεται από τις εξής βασικές αρχές:

- Μια κλινική ερώτηση έχει πολλαπλές κλινικές απαντήσεις. Κάθε απάντηση ορίζεται ξεχωριστά. Οι απαντήσεις αποθηκεύονται σε ξεχωριστό πίνακα με μια σχέση n-προς-1 με τον πίνακα ερωτήσεων.
- Μια κλινική απάντηση συνδέεται με ένα μοντέλο και μία από τις παραμέτρους εξόδου του. Αντιστοιχίζεται είτε σε μία μόνο τιμή παραμέτρου είτε σε ένα συγκεκριμένο εύρος τιμών.
- Οι εκτελέσεις μοντέλων περιλαμβάνουν πληροφορίες σχετικά με τις παραμέτρους εισόδου και εξόδου. Για κάθε παράμετρο εξόδου αποθηκεύονται επίσης η τιμή, οι σχετικές κλινικές ερωτήσεις και οι σωστές απαντήσεις με βάση την τιμή. Κάθε συνδυασμός παραμέτρου-ερώτησης-απάντησης αποθηκεύεται σε ξεχωριστή σειρά.

Εφαρμογή Hyperion: Υλοποίηση και μελλοντικές κατευθύνσεις

Η εφαρμογή Hyperion αποτελεί μια ολοκληρωμένη υλοποίηση πολλών από τις θεμελιώδεις πτυχές και λειτουργίες που περιγράφονται σε αυτή τη διατριβή. Ενοποιεί τις περιγραφόμενες λειτουργίες σε μια συνεκτική, και επεκτάσιμη μονάδα λογισμικού. Το σύνολο τεχνολογιών που επιλέχθηκε για το Hyperion στοχεύει στην ευρωστία, απόδοση και συντηρησιμότητα. Για την αποθήκευση δεδομένων, χρησιμοποιείται το MySQL, ένα ευρέως υιοθετημένο και εξαιρετικά αξιόπιστο σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων. Η αλληλεπίδραση μεταξύ του κώδικα της εφαρμογής και της βάσης δεδομένων διαχειρίζεται από το MyBatis, ένα πλαίσιο που διευκολύνει τη αντιστοίχηση προγραμματιστικών αντικειμένων και σχεσιακών πινάκων βάσης δεδομένων, απλοποιώντας την πρόσβαση και τον χειρισμό δεδομένων. Η διαχείριση του σχήματος βάσης δεδομένων, γίνεται με το εργαλείο ελέγχου έκδοσης Flyway, εξασφαλίζοντας συνέπεια και αξιοπιστία σε διαφορετικά περιβάλλοντα ανάπτυξης.

Το back end του Hyperion είναι κατασκευασμένο χρησιμοποιώντας Spring Boot και Java. Αυτή η επιλογή παρέχει μια ισχυρή βάση εταιρικού επιπέδου, προσφέροντας αξιοπιστία, επεκτασιμότητα για τη διαχείριση αυξανόμενων φορτίων χρηστών, όγκων δεδομένων και προστιθέμενων λειτουργιών, καθώς και φορητότητα μεταξύ πλατφορμών. Αυτό επιτρέπει στην εφαρμογή να λειτουργεί σε διάφορα λειτουργικά συστήματα και περιβάλλοντα διακομιστών χωρίς εξειδικευμένες αλλαγές για το καθένα από αυτά. Το επίπεδο παρουσίασης, έχει αναπτυχθεί χρησιμοποιώντας το React, μια δημοφιλή βιβλιοθήκη JavaScript για τη δημιουργία δυναμικών και διαδραστικών διεπαφών χρήστη. Η αρχιτεκτονική που βασίζεται σε στοιχεία του React προωθεί την επαναχρησιμοποίηση κώδικα και την αρθρωτότητα, επιταχύνοντας την ανάπτυξη και απλοποιώντας τη συντήρηση. Η συνολική δομή κώδικα του Hyperion ακολουθεί μια σαφώς καθορισμένη αρχιτεκτονική τριών επιπέδων: το front end (διεπαφή χρήστη), τους ελεγκτές (που χειρίζονται τα εισερχόμενα αιτήματα και ενορχηστρώνουν την επιχειρηματική λογική) και τις υπηρεσίες (που υλοποιούν την βασική επιχειρηματική λογική) που επικοινωνούν με αποθετήρια (υπεύθυνα για την πρόσβαση σε δεδομένα και την υποβολή ερωτημάτων στη βάση δεδομένων).

Η κύρια μονάδα του Hyperion είναι το Αποθετήριο Μοντέλων. Αυτό το κεντρικό στοιχείο έχει σχεδιαστεί για την ολοκληρωμένη διαχείριση των μοντέλων προσομοίωσης και όλων των σχετικών δεδομένων τους. Αυτό επιτυγχάνεται μέσω ενός δομημένου συνόλου διασυνδεδεμένων πινάκων βάσης δεδομένων που ασχολείται με την περιγραφική πληροφορία, τα αρχεία, τις παραμέτρους, τις ιδιότητες και τις δημοσιεύσεις ενός μοντέλου.

Επιπρόσθετα, το Αποθετήριο Κλινικών Μελετών είναι ένα κρίσιμο χαρακτηριστικό που επιτρέπει την ομαδοποίηση και τη συστηματική διαχείριση πολλαπλών εκτελέσεων μοντέλων. Αυτό είναι ιδιαίτερα ζωτικής σημασίας για τη συλλογική κλινική και στατιστική επεξεργασία, επιτρέποντας στους ερευνητές και τους κλινικούς ιατρούς να αναλύουν αποτελέσματα προσομοίωσης σε ομάδες ασθενών ή διαφορετικά σενάρια θεραπείας. Αυτό το αποθετήριο συνδέει σχολαστικά κλινικές μελέτες με ένα μόνο επιλεγμένο μοντέλο προσομοίωσης και ανωνυμοποιημένα δεδομένα ασθενών, διασφαλίζοντας το απόρρητο, ενώ παράλληλα διευκολύνει την ανάλυση μεγάλης κλίμακας.

Το σύστημα Hyperion ενσωματώνει επίσης το Αποθετήριο Υποστήριξης Κλινικών Αποφάσεων για τη διαχείριση κλινικών ερωτήσεων και απαντήσεων. Αυτές συνδέονται με συγκεκριμένα μοντέλα και τις παραμέτρους εξόδου τους. Αυτή η δυνατότητα επιτρέπει στο σύστημα να ταξινομεί τα αποτελέσματα εκτέλεσης, μεταφράζοντας σύνθετες αριθμητικές και οπτικές εξόδους μιας προσομοίωσης σε εφαρμόσιμες κλινικές γνώσεις που σχετίζονται άμεσα με τη φροντίδα των ασθενών και τη λήψη αποφάσεων.

Πέραν των αποθετηρίων το σύστημα παρέχει τις εξής λειτουργικότητες:

- Εκτέλεση μοντέλων: Οι χρήστες μπορούν να ξεκινήσουν προσομοιώσεις για επιλεγμένα μοντέλα, παρέχοντας δεδομένα εισόδου ειδικά για τον ασθενή.
- Δημιουργία αναφορών: Το Hyperion μπορεί να δημιουργήσει ολοκληρωμένες αναφορές που συνοψίζουν τα αποτελέσματα της προσομοίωσης, προσαρμοσμένες για κλινική ερμηνεία και υποστήριξη αποφάσεων.
- Διαχείριση κλινικών μελετών: Παραγωγή γραφημάτων για την ανάλυση των δεδομένων των πολλαπλών εκτελέσεων που απαρτίζουν μια κλινική μελέτη.

Μέσω της ολοκληρωμένης ανάλυσης και της λεπτομερούς παρουσίασης της εφαρμογής Hyperion, υπογραμμίζεται η κρίσιμη και επείγουσα ανάγκη για μια κατάλληλα σχεδιασμένη, ισχυρή και επεκτάσιμη τεχνολογική υποδομή. Μια τέτοια υποδομή είναι απαραίτητη για την αποτελεσματική γεφύρωση του υπάρχοντος χάσματος μεταξύ των εξαιρετικά πολύπλοκων in silico μοντέλων και της πρακτικής εφαρμογής τους στο εξατομικευμένο ογκολογικό τοπίο. Οι προτεινόμενες λύσεις, ιδιαίτερα η καινοτόμος αρχιτεκτονική και οι λειτουργίες που ενσωματώνονται από την εφαρμογή Hyperion, αντιπροσωπεύουν ένα σημαντικό βήμα προς τα εμπρός στη διευκόλυνση της απρόσκοπτης ενσωμάτωσης αυτών των προηγμένων υπολογιστικών εργαλείων απευθείας στην πολύπλοκη και χρονικά ευαίσθητη διαδικασία λήψης κλινικών αποφάσεων.

Μελλοντικές επεκτάσεις περιλαμβάνουν:

- Τη βελτιστοποίηση για την υποστήριξη πολλαπλών ταυτόχρονων εκτελέσεων, είτε μέσω της χρήσης μεθόδων ταυτοχρονισμού για μεμονωμένες περιπτώσεις, είτε μέσω χρήσης υπολογιστικών συστημάτων που παρέχουν υπηρεσίες όπως διαχείριση εργασιών (Kubernetes), καταγραφή και έλεγχο της κατάστασης του συστήματος (Prometheus, Grafana) σε συνδυασμό με εργαλεία διαχείρισης ουρών (RabbitMQ, Apache Kafka)
- Δημιουργία ξεχωριστού συστήματος για το σχηματισμό αναφορών και τη διεξαγωγή κλινικών μελετών με χρήση μηχανισμών εξαγωγής, μετατροπής και επαναφόρτωσης δεδομένων, σχήματα βάσεων δεδομένων με λιγότερους πίνακες περισσότερων πεδίων για μεγαλύτερη ταχύτητα πρόσβασης και χρήση κατανεμημένων συστημάτων για τη διαχείριση “μεγάλων” δεδομένων (Big Data)
- Ενσωμάτωση του τμήματος παραγωγής μεταδεδομένων στην κύρια εφαρμογή. Χρησιμοποιώντας εξειδικευμένα πλαίσια όπως το Apache Jena, το back end του Hyperion μπορεί να παράξει αυτόματα τα μεταδεδομένα των περιεχομένων του και είτε να τα αποστείλει σε απομακρυσμένα αποθετήρια διασυνδεδεμένων δεδομένων, είτε να παρέχει τις δικές του διεπαφές όπου με τη χρήση ερωτημάτων και APIs θα μπορεί να τα ανταλλάσσει.

- Χρήση τεχνητής νοημοσύνης και αλγορίθμων μηχανικής μάθησης για την αναβάθμιση του μηχανισμού παραγωγής κλινικών ερωτήσεων. Σκοπός είναι να αφορούν άνω της μίας παραμέτρου, ώστε να μπορούν να απαντήσουν σύνθετα ερωτήματα που πιθανόν να αφορούν περισσότερα του ενός μοντέλα, είτε τη δομή ενός υπερμοντέλου, δημιουργώντας κατ' αντιστοιχία "υπερ-ερωτήσεις".

Extended Abstract

This PhD thesis delves into the field of in silico oncology and in silico medicine, focusing on the technologies required to develop integrated Oncosimulators. The latter are considered as advanced digital twins participating in the fight against cancer, designed to improve the way clinicians approach personalized treatment strategies. The main problem faced by this research stems from a fundamental challenge inherent in any simulation model: while the raw input and output data contain clinical knowledge, this is often not readily accessible or understandable to researchers or clinicians. This lack of accessibility significantly hinders the model's direct clinical application in the context of personalized in silico medicine. Addressing the challenge, this thesis proposes and presents the implementation details of a computational infrastructure, which is designed to store and manage the different characteristics of a simulation model (parameters, executable files, basic descriptive information, etc.), including their execution data.

The initial chapters of this work present the fundamental principles governing the design, implementation and functionalities of such an infrastructure. Then, a set of extensions are proposed that aim to enhance the capabilities of the system, and specifically in the handling of semantic data, the definition of clinical questions and relevant answers based on the model characteristics, as well as the management of multiple execution data within the framework of a clinical trial. The practical utility of the entire endeavor, including the Oncosimulators, is presented through the results of European Commission funded research projects. These examples demonstrate how the proposed infrastructure can serve as a key component in the transition of these computational tools from the research laboratory to everyday clinical practice, where they can function as clinical decision support systems. The thesis also presents a prototype implementation of this infrastructure, providing a detailed description of the design techniques and the development tools used (programming languages, etc.), its analysis into discrete, manageable components, and a comprehensive report of various use cases along with their results. The thesis concludes with a concise summary of the findings and outlines future extensions and directions for further research and development in this vital area for the concept of personalized medicine.

In Silico Oncology: Definition and goals

Modern oncology is currently experiencing a shift towards personalized medicine. This shift towards a new approach is driven by an increasingly deeper understanding of the complex mechanisms underlying cancer. Within this dynamic landscape, in silico oncology has emerged as a promising scientific field. It leverages computational modeling and advanced simulation techniques to encompass the multi-layered biocomplexity of cancer progression and response to therapy, thus enabling the development and implementation of personalized treatment strategies. The primary goal of this approach is multifaceted: to significantly enhance the effectiveness of cancer treatments, minimize the side effects experienced by patients, and accelerate the development pace of new drugs. This is achieved by creating virtual representations or digital twins of tumors and simulating their complex interactions with various therapeutic interventions. In essence, in silico oncology extends

the principles of mathematical modeling of biological systems to explain and predict the dynamic behaviors of cancer. This is done by focusing on clinically guided, multiscale simulation models of malignant tumors, taking into account that cancer is a disease that manifests itself at multiple biological scales, from molecular and cellular to organ levels.

The ultimate goal of this science is to use these highly detailed models, after rigorous clinical adaptation and validation, as patient-centered decision support systems and treatment planning tools. This includes the ability to simulate oncology clinical trials, a process that would otherwise be costly, ethically complex and time-consuming if conducted exclusively in traditional laboratories or clinical settings. The efficient execution of these complex simulations is often facilitated by advanced distributed computing infrastructures, such as grid computing.

The genesis of in silico oncology can be traced back to pioneering research conducted at the National Technical University of Athens (NTUA). The original term "in silico radiation oncology" was first introduced in 2002, laying the fundamental conceptual and methodological foundations for what would later evolve into the broader field of in silico medicine. The Oncosimulator, a computational platform developed by the In Silico Oncology and In Silico Medicine Group of the Institute of Communication and Computer Systems (ICCS), quickly became a central, essential component of many completed reserach projects funded by the European Union. These included high-impact initiatives such as ACGT (Advancing Clinico-Genomic Trials on Cancer), ContraCancrum (Clinically Oriented Translational Cancer Multilevel Modeling), CHIC (Computational Horizons In Cancer), MyHealthAvatar (A Demonstration of 4D Digital Avatar Infrastructure for Access of Complete Patient Information) and P-Medicine (From data sharing and integration via VPH models to personalized medicine). Each of these projects had a common goal: to develop, improve and clinically validate advanced multi-scale simulation platforms explicitly designed to support personalized treatment planning for cancer patients. Subsequent projects, such as DR THERAPAT (Digital Radiation Therapy Patient) and BOUNCE (Predicting Effective Adaptation to Breast Cancer to Help Women to BOUNCE back), further expanded the scope of in silico medical approaches, highlighting the potential of this field.

In addition to its research contributions, the field of in silico oncology has also achieved notable academic achievements. Greece, through the initiative of NTUA, is the first country in the world to host a university-level course specifically dedicated to this emerging field. The postgraduate course, entitled "Multiscale Cancer Modeling and In Silico Medicine", was introduced at NTUA in 2014 and continues to be taught, training the next generation of researchers and clinicians in this interdisciplinary field. This academic initiative soon gained international recognition. The Virtual Physiological Human Institute (VPHi), an international organization dedicated to the advancement of in silico medicine, has explicitly highlighted simulation-based healthcare as a strategic direction for future medical developments. At the same time, the Avicenna Alliance, a European industry body, has officially endorsed in silico medicine, recognized Greece and NTUA as its birthplace, and has actively participated in the development of strong regulatory frameworks to

facilitate the widespread adoption and acceptance of in silico clinical trials. These developments highlight the growing momentum and critical role that in silico oncology is poised to play in the future of personalized healthcare.

The Oncosimulator: A digital twin for personalized cancer care

The Oncosimulator is a cornerstone in the growing field of in silico oncology. It is designed to simulate the complex responses of both a cancerous tumor and the healthy normal tissues surrounding it to a wide range of therapeutic strategies. What sets the Oncosimulator apart is the ability to adjust these simulations precisely to each individual patient, utilizing their clinical history, initial imaging data (CT and MRI scans, etc.) as well as molecular and histopathological findings obtained from biopsies. Its primary mission is to achieve the optimal cancer treatment for each patient, not through testing in a living organism, but through computer experiments (thus the name in silico) of a multitude of potential therapeutic approaches. Beyond its utility in supporting clinical decisions, the Oncosimulator also serves a broader purpose: it aims to significantly enhance the scientific understanding of cancer biology, facilitate the design of clinicogenomic trials, and provide a tool for medical education. Based on the digital twin paradigm, the Oncosimulator creates a dynamic, virtual replica of a patient's tumor, allowing for the accurate simulation of tumor progression and its predicted response to various therapies over defined time periods.

The operation of the Oncosimulator is structured around a seven-step process:

1. **Data collection:** This initial step involves obtaining a patient's data that includes a wide range of information, including detailed clinical records, various medical imaging scans (such as CT, MRI, PET), histopathological findings from tissue biopsies, and molecular data (e.g. genomic, proteomic, transcriptomic profiles). The quality and completeness of this data directly affects the accuracy of the simulations.
2. **Data preprocessing:** Once the patient data is acquired, it undergoes a preprocessing stage. During this phase, the heterogeneous data is transformed into a structured format that is directly compatible with the Oncosimulator requirements. This often includes anonymization, de-identification, and error checking to ensure data integrity and patient privacy.
3. **Treatment regimen definition:** In this step, a clinician, based on their experience and the patient's initial diagnosis, defines a set of candidate treatment regimens. These may include various radiotherapy plans, chemotherapy regimens, immunotherapy protocols, or combinations thereof, which will then be tested in the virtual environment.
4. **Simulation execution:** The defined treatment scenarios are executed on powerful, distributed computing resources, typically using grid or cluster architectures. This processing capability is essential, as it allows for the simultaneous and rapid evaluation of multiple treatment options and a wide range of tumor parameter combinations. This significantly reduces computational time, making the simulations clinically useful.

5. **Prediction visualization:** After the simulations are completed, the predicted tumor response and potential toxicological side effects for all simulated scenarios are depicted in detail using numerical diagrams, and 3D static and dynamic visualizations of the tumor image throughout the simulation. The goal is to present complex simulation results in a clear, actionable, and clinically understandable manner.
6. **Clinical evaluation:** Taking into account the visualized predictions, clinicians enter the outcome evaluation phase. They combine extensive medical knowledge, clinical experience, and personalized patient information with the in silico predictions to make an informed decision about the optimal treatment plan. This step highlights the role of the Oncosimulator as a decision support tool, augmenting human expertise rather than replacing it.
7. **Treatment delivery and feedback:** The selected, optimized treatment plan is then delivered to the patient. Data is then systematically collected comparing the Oncosimulator's predictions with actual clinical outcomes and patient responses. This continuous feedback loop is crucial for iteratively improving the Oncosimulator's prediction accuracy and overall performance, ensuring its clinical utility.

The Oncosimulator is based on a "top-down" multiscale simulation strategy, a pioneering approach developed and refined at NTUA. This strategy recognizes that cancer is a multiscale disease, requiring modeling from the molecular and cellular levels to the tissue and organ levels. The fundamental processes of this modeling involve the complex perturbation of radiobiological or pharmacodynamic cell killing parameters, which are dynamically adjusted based on the molecular data of the individual patient. The simulation uses a system for quantifying cell clusters within a discrete grid that precisely covers the anatomical region of interest. Within this grid, cell cycle phase durations and metabolic data are used to determine the path to be followed by the cells of each unit of this grid. Algorithms are then used to simulate macroscopic mechanisms, including dynamic tumor expansion and contraction, the imposition of mechanical boundary conditions on tissues, and the precise effects of various anticancer drugs and radiation on both cancerous and healthy cells.

Mathematically, the Oncosimulator incorporates a range of computational techniques. These include, among others, non-deterministic finite-state automata to model complex cellular behaviors, Monte Carlo techniques to simulate stochastic processes such as radiation deposition and cell death, and differential equations to describe continuous changes in biological parameters over time. From a technological perspective, the Oncosimulator leverages many cutting-edge technologies including dynamic and multidimensional visualization tools for medical data input and simulation predictions, advanced medical image processing techniques for extracting critical anatomical and physiological information, execution acceleration mechanisms (such as grid architectures for distributed computing), and robust systems for the secure and legally compliant transport and storage of highly sensitive biomedical data.

Technological challenges of the transition to personalized medicine

Despite the groundbreaking developments in in silico oncology and its promising contribution to personalized medicine, several significant technological barriers currently prevent its widespread adoption and full clinical implementation. A primary challenge lies in the lack of a robust, scalable, and standardized infrastructure capable of efficiently hosting computational models, performing complex simulations within clinically acceptable time frames, and producing comprehensive, reliable, and clinically actionable reports from the results generated. Managing the vast volume and inherent heterogeneity of patient-related data, ranging from genomic sequences to detailed imaging scans and electronic health records, presents enormous logistical and computational complexity. Furthermore, while computational resources are undoubtedly necessary to perform these complex simulations, especially when fast response times are critical for patient care, access to such high-performance computing (HPC) environments remains a significant constraint for many clinical settings. Furthermore, the inherent diversity of in silico models, each representing different biological scales, physiological processes, or therapeutic approaches, poses significant challenges in terms of their seamless integration and interoperability. Finally, the challenge of converting raw simulation results into standardized, interpretable, and clinically actionable reports for clinicians remains a critical area that requires substantial further development and improvement.

Addressing these central obstacles requires a comprehensive and progressive understanding of the theoretical requirements in several critical areas:

- **Data management:** Establishing rigorous standardization protocols for data formats, nomenclature, and annotations is required to ensure semantic interoperability across different systems and institutions. In addition, developing secure, resilient storage mechanisms and robust management strategies for ever-growing data sets is fundamental, especially while strictly maintaining patient privacy and adhering to stringent data protection regulations (e.g., GDPR, HIPAA). Establishing comprehensive data sharing frameworks that fully comply with the FAIR (Findability, Accessibility, Interoperability, and Reusability) principles is essential to enhance collaborative research initiatives and accelerate scientific discovery.
- **Model integration:** Effective mechanisms are required for the seamless integration of multi-scale models, which inherently represent different levels of biocomplexity (molecular, cellular, tissue, etc.). The development of open platforms that actively support the efficient sharing and practical reuse of in silico models is therefore required to achieve true semantic interoperability between different modeling approaches, mathematical formalisms and simulation tools
- **Computational resources:** In silico simulations require a robust high-performance computing (HPC) infrastructure. This infrastructure must provide scalable resources, specialized software for hosting and running models, as well as advanced analysis tools. Optimization of algorithms and parallelization techniques is essential to

achieve speed and implementation of results in time frames that are meaningful for daily clinical practice.

- **Ethical issues**: Protecting the privacy and security of sensitive health data, as well as limiting bias in AI models, are critical. Transparency and explainability enhance user trust, while ethical frameworks must ensure the balance between scientific progress and individual rights. Finally, accountability and fully informed consent of patients are ethical imperatives.

The current state of personalized oncology

Modern personalized oncology is characterized by the rapid integration of advanced technologies, with Clinical Decision Support Systems (CDSS), such as IBM Watson for Oncology and CSCO AI, leveraging large data sets to provide personalized treatment recommendations. At the same time, systems such as PREDICT focus on more specialized needs, while newer CDSS based on large language models and machine learning enhance the ability to predict and personalize.

In addition, Artificial Intelligence is used in imaging diagnostics, in the creation of detailed patient profiles through genomic, imaging and clinical data, as well as in functional precision medicine, which combines genetic analysis and ex vivo drug testing. Other important technologies include telemedicine and remote monitoring, robotic surgery, advanced electronic health record (EHR) systems with integrated CDSS tools, and specialized mobile applications for oncologists and patients. Artificial Intelligence also enhances drug discovery and reuse, while liquid biopsies offer non-invasive solutions for cancer diagnosis and monitoring. Finally, gene editing technologies such as CRISPR are opening new horizons, although they are still in the research stage.

Despite significant advances, the full clinical utilization of these tools depends on their adaptation to existing workflows, consistency with clinical guidelines, ensuring transparency and reliability, and the adoption of multifactorial approaches that enhance precision and personalization in cancer care.

Conceptual system overview of the proposed infrastructure

The basic idea of the proposed infrastructure integrates relational databases, a web application, and a file storage system. The web application acts as a "shell" around the database, enforcing ACID (Atomicity, Consistency, Isolation, Durability) principles and managing all communications and input data validation. A second layer is dedicated to system security. The architecture emphasizes the standardization of stored descriptive information using an appropriate relational database schema. It is designed to handle various types of data, including model parameters (input and output) and associated files (implementations, documentation, supplementary scenarios). Execution data, such as lists of parameter values per patient and specialized simulation outputs such as graphs and PDF reports, are stored in a combined database and file system for easy retrieval.

The system provides graphical user interfaces (GUI) that offer a range of functions to end users. The main function involves editing the database content, where appropriate

views are created per table, allowing users to perform all Create, Read, Update, and Delete (CRUD) operations on stored objects. Input data is validated against existing database information or model definitions (e.g., preventing biological parameter values outside of a clinically useful range) before being stored.

For model executions, users are provided with a multi-step wizard that presents basic model information, lists input parameters, and prompts for completion to start the run. Input fields can be dynamically generated from the parameter table, ensuring that they are always updated with the latest changes to the model. A second form presents the run results and can generate printable reports in PDF format. Finally, UI tools allow basic analysis of saved run results for any number of model executions. An initial implementation example is presented in the context of the MyHealthAvatar project, including Oncosimulators for nephroblastoma and breast cancer

Metadata generation for in silico models: A theoretical approach

There is a critical need for advanced data management strategies for computational models in in silico oncology, particularly with the rise of hypermodels that incorporate diverse biological scales and aspects of cancer. A theoretical infrastructure is proposed that leverages semantic web technologies for metadata generation and management, adding capabilities to existing data for improved machine understanding and interoperability. This modular infrastructure is designed to work with various software tools, offering flexibility in research environments. It is envisioned to connect to a broader metadata ecosystem to support content publishing, advanced search, and compliance with legal and ethical frameworks for healthcare data. The design incorporates access rights to serve distinct stakeholders, with model authors having upgrade and maintenance privileges, while end users benefit from user-friendly interfaces for running models and retrieving results. The overall system includes seven main modules: the Initial Access Point, the Extract-Transform-Load (ETL) module, two repositories (RDF data and knowledge base), and three front-end applications (comment management, query management, knowledge base management).

Clinical question and answer management

Extending the basic model and execution storage infrastructure, the basic principles for handling clinical questions and their corresponding answers are presented. Through these, a new part of the proposed solution can be created, whose main goal is to enrich the execution results and translate them into valuable clinical knowledge that can be directly applied to personalized medical decisions.

This is achieved by associating the values (individual or ranges) that each parameter of a model can take with the appropriate descriptive information to create a question-answer pair. These pairs will then appear in the execution reports of a model, facilitating the extraction of qualitative conclusions about the course of evolution of a therapeutic regimen, in addition to providing the numerical data of the execution itself.

The relevant system is governed by the following basic principles:

- A clinical question has multiple clinical answers. Each answer is defined separately. The answers are stored in a separate table with an n-to-1 relationship with the question table.
- A clinical answer is associated with a model and one of its output parameters. It is mapped to either a single parameter value or a specific range of values.
- Model executions include information about the input and output parameters. For each output parameter, the value, the associated clinical questions, and the correct answers based on the value are also stored. Each parameter-question-answer combination is stored in a separate row.

Hyperion application: Implementation and future directions

The Hyperion application is a concrete and complete implementation of many of the fundamental aspects and functionalities described in this thesis. It unifies the described functionalities into a coherent, scalable software unit. The set of technologies chosen for Hyperion aims for robustness, performance and maintainability. For data storage, Hyperion uses MySQL, a widely adopted and highly reliable relational database management system. The interaction between the application code and the database is managed by MyBatis, a framework that facilitates the mapping of programming objects to relational database tables, simplifying data access and manipulation. The database schema is managed with the Flyway version control tool, ensuring consistency and reliability across different development environments.

The Hyperion back end is built using Spring Boot and Java. This option provides a strong enterprise-grade foundation, offering reliability, scalability to handle growing user loads, data volumes, and added functionality, as well as portability across platforms. This allows the application to run on a variety of operating systems and server environments without specialized configurations for each of them. The presentation layer is developed using React, a popular JavaScript library for creating dynamic and interactive user interfaces. React's component-based architecture promotes significant code reuse and modularity, speeding up development and simplifying maintenance. Hyperion's overall code structure follows a well-defined three-tier architecture: the front-end (user interface), controllers (which handle incoming requests and orchestrate business logic), and services (which implement core business logic) that communicate with repositories (responsible for accessing data and querying the database).

The main unit of Hyperion is the Model Repository. This central component is designed for the comprehensive management of simulation models and all their associated data. This is achieved through a structured set of interconnected database tables that deal with the descriptive information, records, parameters, properties, and publications of a model.

Furthermore, the Clinical Studies Repository is a critical feature that allows the grouping and systematic management of multiple model executions. This is particularly vital for collaborative clinical and statistical processing, allowing researchers and clinicians to analyze simulation results across patient groups or different treatment scenarios. This

repository meticulously links clinical studies with a single selected simulation model and carefully anonymized patient data, ensuring confidentiality while facilitating large-scale analysis.

The Hyperion system also integrates the Clinical Decision Support Repository for managing clinical questions and answers. These are linked to specific models and their output parameters. This capability allows the system to efficiently categorize run results, translating complex numerical and visual simulation outputs into actionable clinical insights that are directly relevant to patient care and decision-making.

In addition to the repositories, the system provides the following functionalities:

- Model execution: Users can initiate simulations for selected models, providing patient-specific input data.
- Report generation: Hyperion can generate comprehensive reports summarizing simulation results, tailored for clinical interpretation and decision support.
- Clinical study management: Generate graphs to analyze data from multiple executions that comprise a clinical study.

Through the comprehensive analysis and detailed presentation of the Hyperion application, the critical and urgent need for a robust, scalable and appropriately designed technological infrastructure is highlighted. Such an infrastructure is essential to effectively bridge the existing gap between highly complex in silico models and their practical implementation in the personalized oncology landscape. The proposed solution, in particular the innovative architecture and functionalities embodied by the Hyperion application, represent a significant step forward in facilitating the seamless integration of these advanced computational tools directly into the complex and time-sensitive clinical decision-making process.

Future extensions include:

- Optimization to support multiple parallel executions, either through the use of concurrency methods for individual instances, or through the use of computing systems that provide services such as task management (Kubernetes), logging and monitoring of the system state (Prometheus, Grafana) in combination with queue management tools (RabbitMQ, Apache Kafka)
- Creation of a separate system for reporting and conducting clinical studies using data ETL mechanisms, database schemas with fewer tables of more fields for faster access and the use of distributed systems for the management of big data.
- Integration of the metadata generation section into the main application. Using specialized frameworks such as Apache Jena, the Hyperion back end can automatically generate metadata for its content and either send it to remote repositories of interconnected data, or generate its own interfaces where it can exchange it using queries and APIs.
- Use of artificial intelligence and machine learning algorithms to upgrade the mechanism for generating clinical questions. The aim is to involve more than one parameter, so that they can answer complex questions that may involve more than

one model, or the structure of a hypermodel, thereby creating corresponding “hyper-questions”.

1. Introduction

The landscape of oncology has been undergoing a major transformation, shifting from traditional, generic treatment protocols towards personalized medicine. This approach is heavily supported by an increasingly deeper understanding of cancer mechanisms spanning from cellular to organ bio complexity levels, thus revealing the limitations of uniform therapeutic approaches [1]. Therefore transforming therapies from rigid protocols to procedures connected to the unique characteristics of the general clinical picture of individual patients including but not limited to their tumors has become of utmost importance in improving treatment efficacy and minimizing adverse effects [1].

In this context, in silico oncology has emerged as a promising field, harnessing the power of computational modelling and simulation to navigate the complexities of cancer and pave the way for personalized treatment strategies [2]. By creating virtual representations of tumors and their interactions with various therapies, in silico oncology offers the potential to enhance treatment outcomes, lessen side effects, and expedite the discovery of novel drugs [2].

This chapter aims to define in silico oncology, explore essential models such as the Oncosimulator, identify the primary challenges in its application to personalized medicine, discuss the theoretical requirements for a supportive infrastructure, and review the current state of technological advancements in this domain.

1.1. In Silico Oncology – Definition and main objectives

In silico oncology, at its core, is the result of the combination of computer science, applied mathematics and medicine, specifically within the realm of cancer research and treatment [2]. It leverages computer modelling and simulation to delve into the biological mechanisms underlying cancer development, its progression, and its response to therapeutic interventions [2]. The term "in silico", derived from Latin, signifies that these processes are conducted on a computer or through computer simulation [2]. This approach involves the creation of simulation models that visualize tumors and calculate their interactions with different treatment modalities, enabling researchers to predict how cancer will evolve and react to these treatments in a setting that is both risk-free and cost-effective [2].

In essence, in silico oncology extends the application of mathematical modelling to biological systems in order to explain and forecast dynamics specifically within the field of oncology, focusing on the development of clinically driven, multi-scale simulation models of malignant tumors. The objective is to utilize these models, following rigorous clinical adaptation and validation, as patient-specific clinical decision support and treatment planning tools [3]. Standard procedures include the simulation of oncological clinical trials, which would otherwise be prohibitively expensive or time-consuming, utilizing grid computing infrastructures to enhance the efficiency and effectiveness of these simulations [3]. This progression from basic simulations to sophisticated, Artificial Intelligence (AI)-powered frameworks that integrate diverse omics data underscores the ever increasing refinement and evolution of the predictive capabilities of computational models in oncology [4]. The ability of in silico oncology to not only simulate existing biological knowledge but

also to generate novel hypotheses and insights into cancer biology and treatment responses positions it as a vital instrument for pushing the boundaries of cancer research [2].

The formal inception of in silico oncology, and more broadly in silico medicine, is attributed to pioneering work at the National Technical University of Athens (NTUA). In the 2002 article *"In silico radiation oncology: combining novel simulation algorithms with current visualization techniques"*, not only the term *in silico radiation oncology* was introduced but also the conceptual foundations of in silico medicine as a clinically oriented computational discipline were laid [5]

Following this, the Oncosimulator, developed by the In Silico Oncology Group at ICCS–NTUA, became a central component of the ACGT (Advancing Clinico-Genomic Trials on Cancer) project. ACGT was one of the earliest EU-funded integrated projects to explore patient-specific cancer modelling and is recognized as a foundational effort in the development of in silico clinical tools. It demonstrated the use of multiscale models incorporating clinical and genomic data to simulate tumor responses to treatment [6,7].

Subsequent European projects such as ContraCancrum (Clinically Oriented Translational Cancer Multilevel Modelling) [8], CHIC (Computational Horizons in Cancer) [9], and P-Medicine (From data sharing and integration via VPH models to personalized medicine) [10] further developed and clinically validated advanced multiscale simulation platforms in oncology. These efforts aimed to support clinical decision-making and to personalize therapy planning. Later projects like DR THERAPAT (Digital Radiation Therapy Patient) [11], focusing on decision support for personalized radiotherapy, and BOUNCE (Predicting Effective Adaptation to Breast Cancer to Help Women to BOUNCE back) [12], which studied breast cancer patients' resilience trajectories through predictive modelling, expanded the scope and impact of in silico medical approaches .

Academically, Greece became the first country to host an official university-level course dedicated to this emerging field. The doctoral-level course titled *"Multiscale Cancer Modelling and In Silico Medicine"* was introduced in 2014 at NTUA and continues to be taught today [13]. International recognition followed in 2019, as Virtual Physiological Human Institute (VPHi), a key European organization advising the European Commission on the future of in silico medicine [14]. VPHi repeatedly emphasized the importance of simulation-based healthcare as a strategic direction for European biomedical research [15]. Additionally, the Avicenna Alliance, which emerged from the FP7 Avicenna Roadmap initiative, formally endorses the discipline of in silico medicine. This alliance brings together academic and industrial partners to advance regulatory frameworks for in silico clinical trials, working closely with standardization bodies such as ISO and IEC [16].

1.2. The Oncosimulator

The Oncosimulator, a pivotal advanced information system in in silico oncology, is engineered to simulate how tumors and affected normal tissues respond to various therapeutic strategies [17]. These simulations are uniquely tailored to each patient, drawing upon their clinical, imaging, histopathologic, and molecular data [17]. The primary aim of this tool is to optimize cancer treatment for individual patients by conducting computer-

based (in silico) experiments on a range of potential therapeutic approaches [17]. Beyond direct clinical support, the Oncosimulator serves as a platform for deepening our understanding of cancer, designing clinicogenomic trials, and facilitating medical education [17]. Inspired by the digital twin paradigm [18], it aims to simulate the progression of tumors and their response to treatment over time.

The Oncosimulator operates through a structured, seven-step process [19]. First, it involves obtaining the patient's specific data, including clinical details, imaging scans, histopathological findings from biopsies, and molecular information [19]. Second, this collected data undergoes preprocessing to ensure it's in a suitable format for the simulation module [19]. Third, clinicians define one or more candidate therapeutic schemes and/or schedules to be tested [19]. Fourth, these simulations are executed on distributed grid or cluster computing resources, enabling the concurrent evaluation of multiple treatment options under various tumor parameter combinations [19]. Fifth, the predicted tumor response and potential toxicological side effects for all simulated scenarios are visualized using a range of techniques, from simple graphs to virtual reality renderings [19]. Sixth, clinicians carefully evaluate these predictions, drawing upon their medical knowledge, experience, and the logical basis of the results, to decide on the optimal treatment plan for the patient [19]. Finally, the chosen treatment plan is administered, and simultaneously, data comparing the predictions with the actual outcome are collected and used as continuous feedback to refine and improve the Oncosimulator's accuracy [19].

The Oncosimulator is based on the "top-down" multi-scale simulation strategy developed by the In Silico Oncology Group at the National Technical University of Athens [5,20-22]. Provided the Oncosimulator has been validated (retrospectively and prospectively) for a specific application, the imaging, histopathological, molecular, and clinical data of a given patient, following pertinent preprocessing, are introduced into the Tumor and Normal Tissue Response Simulation Module. This module executes the simulation code for a defined candidate treatment scheme (Fig.1-1). The clinician judges the prediction, and if a decision is made to test a further scheme in silico, this is done analogously. Ultimately, the clinician decides on the optimal treatment scheme to administer based on their formal medical education, knowledge, and the predictions of the Oncosimulator.

The most fundamental processes implemented by the Oncosimulator involve using processed molecular data to perturb the radiobiological or pharmacodynamic cell-kill parameters about their population-based mean values. At the core of the simulation approach lies a prototype system of quantizing cell clusters within each geometrical cell of a discretizing mesh, covering the anatomic area of interest. Cell-cycle phase durations and imaging-based metabolism distribution define the quantization equivalence classes considered. Several algorithms have been developed to simulate various macroscopic mechanisms such as tumor expansion or shrinkage and mechanical boundary conditions, as well as the effects of particular drugs (e.g., vincristine, epirubicin, etc.) and radiation on the tumorous and normal tissue under consideration.

From a mathematical standpoint, the Oncosimulator's constituent models utilize several notions and methods, including nondeterministic finite-state automata, the generic Monte Carlo technique, differential equations, general algorithm, and complexity theory. Technologically, numerous current technologies are employed to dynamically and multidimensionally visualize both medical data input and simulation predictions (e.g., through virtual reality platforms), to process medical images [23], to accelerate executions (e.g., through grid architectures), and to securely and legally transfer and store biomedical data (e.g., through pseudonymization).

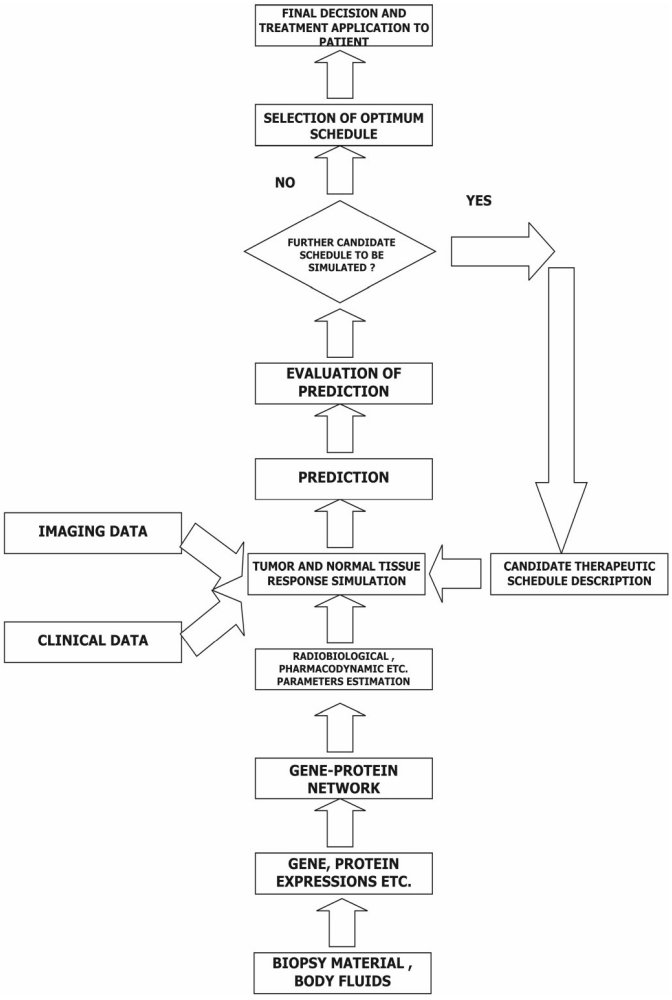


Figure 1-1. Block diagram of the Oncosimulator’s function

The Oncosimulator's capacity to integrate diverse data types and forecast treatment outcomes signifies a substantial advancement towards more comprehensive and individualized modelling in personalized in silico oncology [17]. The structured seven-step process highlights a clear pathway for translating intricate biological data into clinically relevant insights, which is crucial for the eventual integration of in silico oncology into routine clinical practice [19].

1.3. Main technological challenges towards personalize medicine

Despite the remarkable advancements in in silico oncology and tools like the Oncosimulator, a central hurdle remains in the widespread application of personalized medicine within this field: the infrastructure required for hosting models, executing simulations, and generating reports and analyses of these executions. The complexities involved in managing and processing the vast amounts of patient-specific data necessary for in silico oncology present a significant challenge [24].

Robust computational resources are essential to run the intricate simulations within clinically relevant timeframes, yet access to such infrastructure may be limited [4]. Furthermore, the integration of diverse models that represent different biological scales and therapeutic modalities poses a considerable difficulty [3]. Finally, the generation of standardized and clinically meaningful reports from the outputs of these simulations remains a critical area needing development [24]. The absence of a standardized and scalable infrastructure acts as a major impediment, hindering the broad adoption of in silico oncology in personalized medicine [24].

Overcoming these challenges is vital to translate the immense potential of computational models into tangible benefits for individuals battling cancer. The infrastructural needs extend beyond mere computational power, encompassing sophisticated data management systems, interoperable model and data repositories, and user-friendly platforms that enable clinicians to access and interpret simulation results effectively [24].

This multifaceted challenge underscores the interdisciplinary nature of building a robust foundation for enhancing personalized medicine by utilizing the outcomes of in silico oncology. Addressing the central hurdle requires a comprehensive understanding of the theoretical requirements for building such infrastructures These requirements span several critical domains, including data management, model integration, computational resources, and ethical considerations.

Effective data management is paramount, necessitating the integration of heterogeneous data from diverse sources such as clinical records, medical imaging, and various omics platforms (genomics, proteomics, etc.) [18]. To ensure interoperability, the standardization of data formats, nomenclature, and annotations is crucial [24]. The secure storage and management of these large-scale datasets, while upholding patient privacy and data protection regulations, are also fundamental [24]. Furthermore, the establishment of data sharing frameworks that adhere to the FAIR principles (Findability, Accessibility, Interoperability, Reusability) is essential for advancing collaborative research [24]. The quality and annotation of datasets used to train and validate in silico models directly impact their reliability and predictive power [24]. Finally, maintaining the traceability and provenance of data throughout the modelling and simulation pipeline ensures the integrity of the results [25].

Model integration forms another cornerstone of the required infrastructure. Frameworks are needed to integrate multiscale models that represent the intricate hierarchy of biological organization, from molecules to entire systems [3]. Achieving semantic interoperability between different modelling approaches and simulation tools is vital for

creating a cohesive and usable infrastructure [26]. Platforms that support the collaborative development, sharing, and reuse of in silico models can accelerate progress and prevent redundancy [26]. Standardized formats and interfaces for model exchange and composition will further enhance the modularity and flexibility of the infrastructure [27].

The computational demands of in silico oncology necessitate robust computational resources. Access to high-performance computing (HPC) infrastructure, including both clusters and cloud-based solutions, is essential for running complex simulations [4]. The infrastructure must provide scalable computing resources to handle the computational intensity of large-scale simulations [26]. Specialized software and tools tailored for model development, simulation execution, and data analysis are integral components [24]. Finally, the implementation of efficient algorithms and parallelization techniques is crucial for optimizing simulation performance and achieving results within clinically relevant timeframes [19].

Ethical considerations are paramount in the development and deployment of infrastructures for personalized in silico oncology. Ensuring the privacy and security of sensitive patient health information is non-negotiable [24]. Addressing potential biases embedded within data and models is critical to prevent the exacerbation of health disparities [24]. Transparency and explainability of AI-driven models are vital for fostering trust among both clinicians and patients [25]. Ethical frameworks governing data sharing and reuse must strike a balance between advancing research and safeguarding individual rights [24]. Clear lines of accountability and responsibility must be established for decisions informed by in silico predictions [28]. Lastly, obtaining informed consent for the collection and use of patient data in in silico research is an ethical imperative [28].

1.4. State of the Art

The current state of the art in technological assistance for personalized oncology reveals a significant integration of advanced technologies to aid physicians in providing tailored care. Clinical Decision Support Systems (CDSS) have emerged as valuable tools, exemplified by Watson for Oncology (WFO), an AI-based system that offers evidence-informed treatment recommendations drawing from the National Comprehensive Cancer Network (NCCN) guidelines and a vast body of medical literature [29]. WFO supports clinicians in making decisions across various cancer types, including lung, colon, and breast cancer, by analyzing patient data and providing ranked treatment options with links to supporting evidence [29]. Similarly, the CSCO AI system, developed in China, based on national guidelines and extensive data, focuses on providing personalized treatment suggestions specifically for breast cancer [29]. Other CDSS, such as PREDICT, offer more focused support, helping patients and clinicians understand the potential survival benefits of different treatments for early invasive breast cancer [30]. The development of AI-CDSS leveraging advanced models like those powered by ChatGPT demonstrates the continuous evolution of these tools towards enhanced data processing and prediction accuracy, as seen in the context of breast cancer recurrence [31].

Beyond CDSS, a multitude of other technologies are being employed to assist physicians in delivering personalized oncology care. AI-driven diagnostics are increasingly utilized to analyze medical images, such as X-rays, CT scans, and MRIs, enabling more accurate detection of cancer [32]. Functional precision medicine represents a novel approach that combines genetic testing with the direct screening of drug efficacy on living tumor cells derived from a patient, offering a personalized strategy particularly valuable when actionable mutations are not readily identifiable through genomics alone [1]. Comprehensive genomic profiling has become a cornerstone of personalized oncology, providing detailed insights into the molecular characteristics of a patient's tumor to guide targeted therapies [1]. AI is also being applied to patient profiling, analyzing complex datasets encompassing genomics, imaging, and electronic health records to tailor cancer therapy to individual needs [25]. The integration of telemedicine and remote monitoring through apps and platforms allows for remote consultations and the tracking of treatment toxicities, improving patient convenience and potentially treatment outcomes [33]. Robotic surgery offers surgeons enhanced precision and control in complex cancer procedures [34]. Electronic Health Record (EHR) systems, such as iKnowMed, are being equipped with integrated decision support tools to provide point-of-care guidance to oncologists [35]. Mobile applications like ONCOassist provide oncologists with immediate access to tools for risk stratification, drug dosage calculations, and chemotherapy protocol planning [35]. Furthermore, AI is playing a crucial role in drug discovery and repurposing efforts, accelerating the identification of new therapeutic candidates and potential new uses for existing medications [4]. The development of liquid biopsies offers a less invasive method for detecting cancer signals in blood, facilitating both diagnosis and monitoring [33]. Finally, gene editing tools like CRISPR hold future promise for the precise correction of genetic aberrations underlying cancer [36]. While these advancements demonstrate a clear trend towards leveraging AI and machine learning to manage the complexity of cancer, the effectiveness and integration of tools like CDSS are still influenced by factors such as guideline variations and the need for further optimization within clinical workflows [37]. Functional precision medicine offers a valuable complementary strategy to genomics-based approaches, highlighting the importance of multifaceted approaches in tailoring cancer treatment [1].

1.5. Chapter overview

Following this introduction of basic principles chapter 2 presents the fundamental characteristics for a simulation model hosting infrastructure. It provides the overall architecture, a detailed description of the individual components, and portrays an initial implementation paradigm in the context of the MyHealthAvatar project [38], combined with two of the Oncosimulator branches, Wilms tumor and breast cancer. Chapter 3 proposes a modular extension of chapter 2’s framework, that can extract, store and query semantic metadata related to the contained simulation models, their characteristics and their execution results.

In chapter 4 we explore the next level of simulation model technological integration by demonstrating the capabilities of the Oncosimulator in the context of Clinical Decision Support Systems (CDSS). Through the example implemented for the P-Medicine project [17], we show the integration procedure implementation details and the results as provided in the overall system output. Furthermore, in chapter 5, we deal with the notion of standardization of clinical questions and their answers. Through a second extension of chapter 2's outcome, we enrich the functionality of our proposed work by the creation, storage, definition and usage of clinical answers. By including the pertinent information in model execution results, both main stakeholders (clinicians and modelers) can benefit from either the increased information and its delivery pace to assist in a more person-oriented medical decision-making, or the further textual classification of multiple model execution results, which through a proper presentation and calculation layer can provide useful insights for model adaptation and validation.

Chapter 6 presents the Hyperion application, which implements numerous aspects of the work mentioned in previous chapters. Using the previous projects as an example, it encloses all the described functionality in a single cohesive but scalable and extendable unit. We describe aspects such as the technical stack, the implementation details, the different internal modules in all three layers (front-end, back end and database schema). Chapter 7 lists the usage workflows of Hyperion, along with the application's results for each of its functionalities. Our work is concluded with the final discussion in chapter 8.

1.6. References

1. Redefining Personalized Cancer Treatment | FIU Research ..., accessed on April 6, 2025, https://news.fiu.edu/2025/redefining-personalized-cancer-treatment

2. National Technical University of Athens. Research Prof. G. Stamatakos is interviewed by the Chinese and international AI technology community platform THE YUAN on the globally acknowledged emergence of in silico (computational) medicine in ECE, NTUA, Greece (2022), accessed on April 6, 2025, https://www.ece.ntua.gr/en/article/530

3. IN SILICO MEDICINE DEFINITION & HISTORY | In Silico Onc Med, accessed on April 6, 2025, https://www.in-silico-oncology.iccs.ntua.gr/about-1

4. Bioscience, C., (2025) In Silico Models in Oncology: Validating AI-Driven Predictive Frameworks - Crown Bioscience, accessed on April 6, 2025 https://blog.crownbio.com/in-silico-models-in-oncology-validating-ai-driven-predictive-frameworks

5. Stamatakos, G. S., Dionysiou, D. D., Uzunoglu, N. K., & Nikita, K. S. (2002). In silico radiation oncology: Combining novel simulation algorithms with current

visualization techniques. *Proceedings of the IEEE*, 90(11), 1764–1777. https://doi.org/10.1109/JPROC.2002.804676

6. E. C. Georgiadi et al., "Multilevel cancer modeling in the clinical environment: Simulating the behavior of Wilms tumor in the context of the SIOP 2001/GPOH clinical trial and the ACGT project," 2008 8th IEEE International Conference on BioInformatics and BioEngineering, Athens, Greece, 2008, pp. 1-8, doi: 10.1109/BIBE.2008.4696759.

7. Stamatakos, G. S., Kolokotroni, E. A., Dionysiou, D. D., Georgiadi, E. C., & Desmedt, C. (2014). Exploiting clinical trial data for multiscale cancer modeling: The ACGT Oncosimulator validation studies. *IEEE Journal of Biomedical and Health Informatics*, 18(3), 708–718. https://doi.org/10.1109/JBHI.2013.2290381

8. Marias, K. *et al.* (2009). Clinically Oriented Translational Cancer Multilevel Modeling: The ContraCancrum Project. In: Dössel, O., Schlegel, W.C. (eds) World Congress on Medical Physics and Biomedical Engineering, September 7 - 12, 2009, Munich, Germany. IFMBE Proceedings, vol 25/4. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-03882-2_564

9. Stamatakos G, Dionysiou D, Misichroni F, Graf N, van Gool S, Bohle R, Dong F, Viceconti M, Marias K, Sakkalis V, Forgo N, Radhakrishnan R, Byrne H, Guiot C, Buechler P, Neri E, Bucur A, de Bono B, Testi D, Tsiknakis M. Computational Horizons In Cancer (CHIC): Developing Meta- and Hyper-Multiscale Models and Repositories for *In Silico* Oncology - a Brief Technical Outline of the Project. Proc 2014 6th Int Adv Res Workshop In Silico Oncol Cancer Investig (2014). 2014 Nov;2014:10.1109/iarwisoci.2014.7034630. doi: 10.1109/iarwisoci.2014.7034630. PMID: 34541587; PMCID: PMC8444280.

10. Marés J, Shamardin L, Weiler G, Anguita A, Sfakianakis S, Neri E, Zasada SJ, Graf N, Coveney PV. p-medicine: A Medical Informatics Platform for Integrated Large Scale Heterogeneous Patient Data. AMIA Annu Symp Proc. 2014 Nov 14;2014:872-81. PMID: 25954394; PMCID: PMC4419910.

11. European Commission. (n.d.). DR THERAPAT: Decision Support System for Therapy in Head and Neck Cancer. *CORDIS EU Research Results*. https://cordis.europa.eu/project/id/600852

12. H. Kondylakis *et al*., "Developing a Data Infrastructure for Enabling Breast Cancer Women to BOUNCE Back," *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*, Cordoba, Spain, 2019, pp. 652-657, doi: 10.1109/CBMS.2019.00134.

13. National Technical University of Athens. (2014). Course: Multiscale Cancer Modelling and In Silico Medicine. *School of Electrical and Computer Engineering*. https://www.ece.ntua.gr/gr/article/477

14. National Technical University of Athens. Research Prof. Georgios Stamatakos elected a member of the Board of Trustees of the international Virtual Physiological Human Institute (VPHi) (2019). https://www.ece.ntua.gr/gr/article/395

15. Virtual Physiological Human Institute (VPHi). (n.d.). *Official website*. https://www.vph-institute.org

16. Avicenna Alliance. (n.d.). The Avicenna Roadmap and the future of in silico medicine. *Official site*. https://www.avicenna-alliance.com/about/the-avicenna-project.html

17. G. S. Stamatakos *et al*., "The "Oncosimulator": a multilevel, clinically oriented simulation system of tumor growth and organism response to therapeutic schemes. Towards the clinical evaluation of in silico oncology," *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Lyon, France, 2007, pp. 6628-6631, doi: 10.1109/IEMBS.2007.4353879.

18. Meyerheim M, Panagiotidou F, Georgiadi E, Soudris D, Stamatakos G and Graf N (2025) Exploring the *in silico* adaptation of the Nephroblastoma Oncosimulator to MRI scans, treatment data, and histological profiles of patients from different risk groups. *Front. Physiol.* 16:1465631. doi: 10.3389/fphys.2025.1465631

19. M. Blazewicz, E. C. Georgiadi, J. Pukacki and G. S. Stamatakos, "Development of the p-medicine oncosimulator as a parallel treatment support system," *Proceedings of the 2014 6th International Advanced Research Workshop on In Silico Oncology and Cancer Investigation - The CHIC Project Workshop (IARWISOCI)*, Athens, Greece, 2014, pp. 1-5, doi: 10.1109/IARWISOCI.2014.7034641.

20. D. D. Dionysiou, G. S. Stamatakos, N.K. Uzunoglu, K. S. Nikita, A.Marioli, "A four-dimensional simulation model of tumor response to radiotherapy in vivo: parametric validation considering radiosensitivity, genetic profile and fractionation," *Journal of Theoretical Biology* 230 (2004) 1–20

21. G. S. Stamatakos, V. P. Antipas, N. K. Uzunoglu *"*A Spatiotemporal, Patient Individualized Simulation Model of Solid Tumor Response to Chemotherapy *in Vivo*: The Paradigm of Glioblastoma Multiforme Treated by Temozolomide," *IEEE Trans. Biomedical Engineering*, vol. 53, No.8, pp. 1467-1477, 2006.

22. G. Stamatakos, D. Dionysiou, and N. Uzunoglu, “In Silico Radiation Oncology: A Platform for Understanding Cancer Behavior and Optimizing Radiation Therapy Treatment,”in M.Akay Ed. “*Genomics and Proteomics Engineering in Medicine and Biology*,” Wiley-IEEE Press, 2007

23. K. Marias, Th. Margaritis, F. Zacharopoulou, E. Georgiadi, T.G. Maris, G. Tollis, C.P.Behrenbruch, “Multi-level analysis and information extraction considerations for validating 4D models of human function,” ,” in *Proc. 2nd International Advanced Research Workshop on In Silico Oncology*, Kolympari, Chania, Greece, 25-26 September 2006, Eds. K. Marias and G. Stamatakos, pp 48-50, http://www.ics.forth.gr/bmi/2nd-iarwiso/

24. How Data Science and Bioinformatics Services Are Revolutionizing Oncology Research, accessed on April 6, 2025, https://blog.crownbio.com/how-data-science-and-bioinformatics-services-are-revolutionizing-oncology-research

25. www.vph-institute.org, accessed on April 6, 2025, https://www.vph-institute.org/upload/vph-fet-final-roadmap-1_519244713c477.pdf

26. Servant N, Roméjon J, Gestraud P, La Rosa P, Lucotte G, Lair S, Bernard V, Zeitouni B, Coffin F, Jules-Clément G, Yvon F, Lermine A, Poullet P, Liva S, Pook S, Popova T, Barette C, Prud’homme F, Dick J-G, Kamal M, Le Tourneau C, Barillot E and Hupé P (2014) Bioinformatics for precision medicine in oncology: principles and application to the SHIVA clinical trial. *Front. Genet.* **5**:152. doi: 10.3389/fgene.2014.00152

27. Hunter P, Coveney PV, de Bono B, Diaz V, Fenner J, Frangi AF, Harris P, Hose R, Kohl P, Lawford P, McCormack K, Mendes M, Omholt S, Quarteroni A, Skår J, Tegner J, Randall Thomas S, Tollis I, Tsamardinos I, van Beek JH, Viceconti M. A vision and strategy for the virtual physiological human in 2010 and beyond. Philos Trans A Math Phys Eng Sci. 2010 Jun 13;368(1920):2595-614. doi: 10.1098/rsta.2010.0048. PMID: 20439264; PMCID: PMC2944384.

28. Herington J, McCradden MD, Creel K, Boellaard R, Jones EC, Jha AK, Rahmim A, Scott PJH, Sunderland JJ, Wahl RL, Zuehlsdorff S, Saboury B. Ethical Considerations for Artificial Intelligence in Medical Imaging: Data Collection, Development, and Evaluation. J Nucl Med. 2023 Dec 1;64(12):1848-1854. doi: 10.2967/jnumed.123.266080. PMID: 37827839; PMCID: PMC10690124.

29. Wang L, Chen X, Zhang L, Li L, Huang Y, Sun Y, Yuan X. Artificial intelligence in clinical decision support systems for oncology. Int J Med Sci. 2023 Jan 1;20(1):79-86. doi: 10.7150/ijms.77205. PMID: 36619220; PMCID: PMC9812798.

30. Mazo C, Kearns C, Mooney C, Gallagher WM. Clinical Decision Support Systems in Breast Cancer: A Systematic Review. *Cancers*. 2020; 12(2):369. https://doi.org/10.3390/cancers12020369

31. Lin T, Chung H, Jian M, Chang C, Perng C, Liao G, Yu J, Dai M, Yu C, Shang H. An Advanced Machine Learning Model for a Web-Based Artificial Intelligence–Based Clinical Decision Support System Application: Model Development and Validation Study. J Med Internet Res 2024;26:e56022. https://www.jmir.org/2024/1/e56022. Doi: 10.2196/56022

32. 10 Examples of Clinical Decision Support Systems Applications - Multimodal.dev, accessed on April 6, 2025, https://www.multimodal.dev/post/examples-of-clinical-decision-support-systems-applications

33. 6 experts reveal the technologies set to revolutionize cancer care, accessed on April 6, 2025, https://www.weforum.org/stories/2022/02/cancer-care-future-technology-experts/

34. Emerging Technology and Tools Revolutionizing Oncology Care - AMN Healthcare, accessed on April 6, 2025, https://www.amnhealthcare.com/blog/physician/locums/emerging-technology-and-tools-revolutionizing-oncology-care/

35. Oncology Clinical Management Technology | McKesson, accessed on April 6, 2025, https://www.mckesson.com/specialty/oncology-clinical-management-technology/

36. Ho D, Quake SR, McCabe ERB, Chng WJ, Chow EK, Ding X, Gelb BD, Ginsburg GS, Hassenstab J, Ho CM, Mobley WC, Nolan GP, Rosen ST, Tan P, Yen Y, Zarrinpar A. Enabling Technologies for Personalized and Precision Medicine. Trends Biotechnol. 2020 May;38(5):497-518. doi: 10.1016/j.tibtech.2019.12.021. Epub 2020 Jan 21. PMID: 31980301; PMCID: PMC7924935.

37. Suwanvecho S, Suwanrusme H, Jirakulaporn T, et al. Comparison of an oncology clinical decision-support system's recommendations with actual treatment decisions. *J Am Med Inform Assoc*. 2021;28(4):832-838. doi:10.1093/jamia/ocaa334

38. Spanakis E, Kafetzopoulos D, Yang P, Marias K. MyHealthAvatar: personalized and empowerment health services through internet of things technologies. *Wireless*

Mobile Communication and Healthcare (Mobihealth), 2014 EAI 4th International Conference on Wireless Mobile Communication and Healthcare – Transforming Healthcare Through Innovations in Mobile and Wireless Technologies. Athens, Greece, 2014:331–4. doi:10.4108/icst.mobihealth.2014.257500.

2. Basic Principles for Model Hosting Infrastructures

This chapter delves into the foundational principles and architectural design of a model hosting infrastructure. It outlines the conceptual underpinnings of the system, emphasizing the integration of relational databases and file storage for robust data management. Furthermore, the chapter details the key components of the infrastructure, including its core based on the Model-View-Controller (MVC) pattern, the persistence layer responsible for data storage, the graphical user interfaces facilitating user interaction, the Application Programming Interface (API) enabling data exchange, and the security measures implemented to ensure data integrity and user authentication. By listing the interrelations between these components, this chapter provides a comprehensive overview of the infrastructure's architecture. Finally, it introduces the MyHealthAvatar [1] paradigm and two specific Oncosimulator models (Wilms tumor [2,3] and breast cancer [4,5]) as illustrative examples of the infrastructure's application and versatility in handling diverse simulation tools and user needs.

2.1. Conceptual system overview

The primary idea, on which the present infrastructure as well as the paradigm prototype have been built, is based on a combination of relational databases, a web application and a file storage system. Aiming to ensure the enforcement of the Atomicity, Consistency, Isolation, Durability (ACID) principles and the handling of properly formed data, the database set is “wrapped” by the web application, thereby acting as a “shell”, which includes all communication methods and procedures between the infrastructure and the stakeholder groups, as well as all input data validation controls. Around this shell lays a second one, related to the security of the entire system.

To achieve a level of standardization of the stored data, a suitable schema had to be selected for the relational databases. According to this schema, for each stored object, its individual characteristics and structural elements are placed in different tables. One table handles one specific feature for all stored objects. To represent a stored object in full, these tables are linked together with one-to-many or many-to-many relations.

Figure 2-1 shows the high-level conceptual diagram of the envisaged system. This initial blueprint has been appropriately modified and extended, taking into account the types of the prospective users and their particular needs for communicating with the infrastructure. These modifications have been developed to satisfy a set of user requirements, which was introduced during the early stages of the infrastructure implementation.

2.2. Components and Interrelations

Infrastructure “core”. The infrastructure backbone is based on the model-view-controller (MVC) [6] architectural pattern. Each data storage unit – in this case, a database table – is assigned to a model. All available functions for model manipulation are accessed by the users through the controller methods. These include modifications of the database

contents, tool executions, application programming interface (API) calls, return of corresponding results, and communication with the individual engines (execution, report/analysis, etc.). The controller methods are accessible via different URLs. This dictates the operation of the views that correspond to the graphical user interfaces (GUIs). The views are provided to the users for the subset of the model-manipulating functions, which require manual input, such as uploading a tool. The application of the MVC pattern to the conceptual system diagram of Figure 2-1 results in the more specific infrastructure architecture depicted in Figure 2-2.

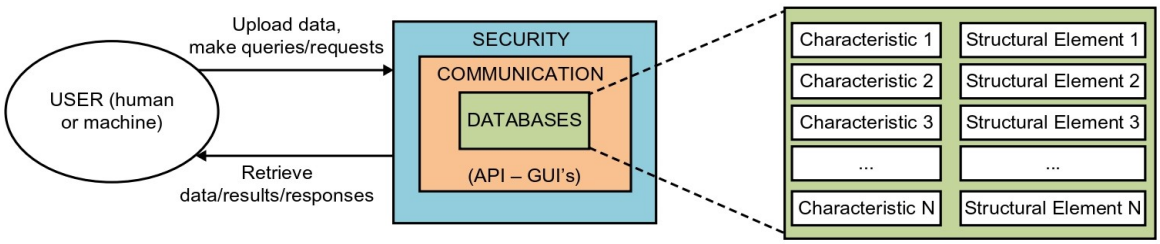


Figure 2-1. Conceptual diagram of the proposed architecture and database schema.

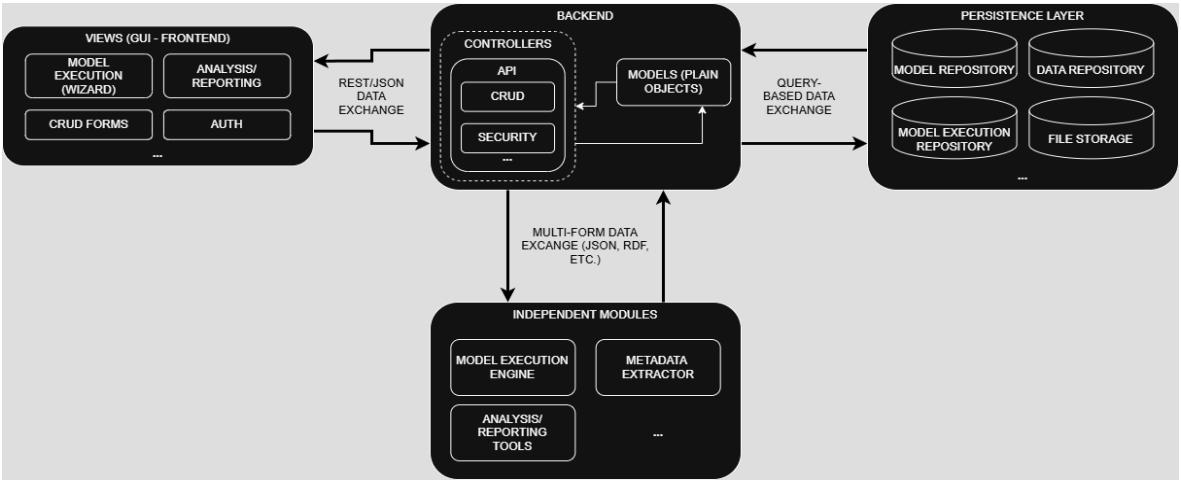


Figure 2-2. Architecture infrastructure

Peristence Layer. This is the part where the information that is managed by the infrastructure resides. It consists of one or more relational databases, depending on the desired implementation details, as well as a designated folder system within the OS for file storage. It includes the information that pertains to the simulation models and additional software tools that may assist in the model execution. The entity–relationship schema, which is adopted, groups and separates the various characteristics and structural components of a model, in a manner that is unaffected by the model format, functionality, or simulated condition. Any data used in model executions are also placed in another database or a well defined set of tables, which may also be used as storage for execution results. Depending on the desired implementation scale, additional tables, associated with the operation of other infrastructure parts, can be grouped on a separate database or be included into the data

repository on an ad hoc basis. A typical example is the use of a table as a log for model executions, which can be fed by the model execution engine.

To create the schema for the repository databases, a number of assumptions are made in order to achieve the conceptual separation of a model's characteristics:

- Each model has basic descriptive information, which is stored separately from its implementations and other basic features.
- One or more properties further describe and/or classify a model. These properties provide additional information about the model and may relate to its operating principle.
- The descriptive information or a property and its value per model are stored in different tables. A property value is connected to the model it characterizes using a many-to-many relationship with a dedicated table, thereby facilitating the reuse of the property in several models.

Furthermore, each model has a number of parameters, which are divided into input parameters and output parameters. It is also accompanied by a set of files that include the model implementations, documentation and instructions, supplementary scripts, etc. The database holds all the descriptive information, whereas the actual files (data) are stored internally in a file-based repository or a designated folder system within the operating system. The execution data are stored as lists of parameter values for a given model, and they can appear in the form of files, especially in the case of execution results. These lists are currently made of value sets per subject. Two different values for a given parameter can refer to the same subject if the parameter refers to a biological marker measured more than once overtime (e.g, a subject's white blood cell count). In addition to the value sets, any specialized simulation output (graphs, constructed pdf reports, images, etc.) are stored in the layer's file storage, where they can be retrieved as necessary.

Graphical user interfaces. A set of functionalities are available to the end users, via a graphical environment. The main one involves editing the contents of databases. Using the MVC models as blueprints, suitable views can be created per table, containing data fields that match the table columns. Therefore, the user can perform all create, read, update, and delete (CRUD) operations on the stored objects. Input data are validated before committing a transaction according to pertinent information already located in the databases or defined in the model implementations (e.g. preventing the user from setting a biological parameter value for a model execution out of its permitted range, which is stored in the parameter table).

As it pertains to model executions, in the simplest of cases, the user can be provided with a view for a particular model that can present its basic information, a list of the input parameters, and prompt the user to complete them and start the execution. The input fields for these parameters can be either explicitly included in the view if it is preconstructed with handwritten code or received dynamically from the database with a query to the parameter table asking for the input parameters of the model. This way the form is created on-the-fly thus being always automatically updated with the latest changes of the model. The view is responsible for presenting the results of an execution to the user. It also provides the ability

to produce printable reports (e.g, in pdf format). For multiple models directed toward the same disease or addressing different aspects and variations of a given disease (e.g, showing all models addressing one specific type of cancer), the view can take the form of a wizard, which enables the user to initially determine the disease or variation they want to experiment with and then choose the desired model from a suitably selected set. To further distinguish this model execution functionality from the rest of the infrastructure GUIs, this wizard-based environment will from now on be referred to as the model execution module.

Finally, since the capability exists to store any number of model executions, a set of UI tools is provided with which the user can ran some basic analysis of the stored execution results and produce pertinent graphs. These functionalities can help the modeler in their adaptation and validation procedures.

API. This is the infrastructure gateway with respect to data exchanges with the GUI as well as with third-party applications and the independent modules. The representational state transfer (REST) architectural style is used, and the exchanged data are in JavaScript Object Notation (JSON) format. Similar to the previous component, database CRUD capabilities are offered, albeit on a larger scale (actions on multiple or all entries of a table or on a query-determined set of them), by using all the classic HTTP methods (GET, POST, PUT, DELETE). The set of the included web services is accessible by a URL system, the form of which is determined through the API implementation software. For simplicity, a common form is suggested with the URL system of the controllers.

Security. The infrastructure is accessed from two different types of users (humans and third-party applications). This dictates the required security measures that provide authentication and authorization services. For physical users, a username/password system is utilized, which also supports granting specific access rights to certain parts of the infrastructure to distinct user group profiles (e.g, clinician, modeler, etc.). For applications requiring access to the infrastructure, appropriate web protocols are used. Common solution is OAuth2.0, [7] which is used by companies like Google and Facebook and/or JWT or API key solutions to secure the application's APIs. External security mechanisms can also be used in cases where the infrastructure is part of a larger platform (e.g, a trusted third party can handle all the identification and the chosen user group rights policies and provide access via a single sign-on mechanism). In this case, all that is required is the notification/alteration of the controllers to allow the requested functionality and communicate with the established mechanism. For a standalone version, the security features are included as parts of the infrastructure. Data about individual users, user groups, and access rights can be stored in tables in the database layer. The login and sign-up procedures are performed through the views. Finally, the API with the proper additions to its pool of web services can undertake the identification of external applications before any remote data exchanges take place.

Independent modules. To enhance the overall operation of the infrastructure, specialized modules are used occasionally, which perform specific processes pertaining to either the transformation of the stored data or performance optimization.

The model execution engine is tasked with the management of all requested and pending model executions. Any requested model execution is treated as a separate task and

handled properly, while its status is monitored. The relevant information is stored in real time in the database layer, from where it can be retrieved later for reporting reasons, statistical analyses or any clinical consideration. The need for such a component emerges implicitly from the diversity of the stored models since, depending on the disease, the biological level whose functions are simulated and the methods of development used for the model creation, it is possible that model execution times can vary, reaching from a few seconds up to hours.

The analysis/reporting tool has a double role. First, in an effort to facilitate the clinician's work, it can communicate with the database layer, retrieve the results of an execution and create a visual and a downloadable pdf report containing these results. The tool can also be beneficial for the modeler, as a model can be chosen, and correlation graphs for its variables can be created, using all the available execution data. This can help the modeler with the adaptation and/or validation procedures.

As a theoretical addition, a metadata creation module can be also be considered (simulation model semantic annotator). Such modules can assign metadata to the stored models, and/or any pertinent data, in order to achieve semantic linkage. As a bare minimum, a semantic URL can be stored along with each entity in a specific database field.

2.3. The MyHealthAvatar paradigm

Based on the blueprint of Figure 2-2, a working prototype application, named IAPETUS, had been implemented in the context of the MyHealthAvatar project. The MyHealthAvatar project was a feasibility study, aiming to collecting personal data and utilizing them to advance healthier lifestyles through various processes, including disease prevention. This led to a scenario-based design, including use cases and workflows that implement them. The initial objective was to create a repository that would host the models to be developed and used for the project purposes, along with the necessary execution data.

2.3.1. Implementation stack

The majority of the infrastructure components were implemented using the Python programming language through various software developing tools. More specifically, the Django framework [8] was used for implementing the MVC mechanism of the infrastructure "core", Tastypie [9] was used for the API RESTful web services, and Celery [10] was chosen as a task creation mechanism for the model execution engine. The engine operated by combining Celery with a message broker (RabbitMQ [11]) and a separate result back-end database (MongoDB [12]). Security was handled by the inherent Django user authentication system and the Django OAuth Toolkit, [13] which was used to verify external applications requesting access, using the OAuth2.0 standard. Finally, the relational databases were built using MySQL.

It should be noted that according to the Django interpretation of the MVC pattern, a view is used as a callback function for a particular URL. This means that, in an attempt to separate content from presentation, a view "determines which data are presented to the user, not how they are presented".[8] Data presentation is handled by the so-called templates,

which, in the majority of cases, are implemented as html pages that are sent to the user's screen. Therefore, a more accurate description of the approach would be model–template–view (MTV). This means that for this initial implementation of the component description mentioned in section 2.2, the MVC controllers were the Django views and the MVC views were the Django templates. Following the principles of the MVC – or in this case, MVT – pattern, it was possible to capitalize on the potential for scalability offered by Django and include the model execution module.

2.3.2. The Wilms tumor Oncosimulator

The primary example was the nephroblastoma (Wilms tumor) Oncosimulator [2,3], developed by the In Silico Oncology and In Silico Medicine Group, Institute of Communication and Computer Systems, National Technical University of Athens. It is a model that simulates tumor response to preoperative chemotherapy treatment with actinomycin and vincristine. It is developed with discrete mathematics, following a top–down approach. The model starts from the macroscopic high biocomplexity level (imaging data) and proceeds toward lower biocomplexity levels. The macroscopic anatomic region of interest is either manually or semi-automatically annotated by the clinicians on MRI imaging sets acquired at the time of diagnosis. A virtual cubic mesh is used for the discretization of the area of interest (tumor) of which the elementary cube is termed geometrical cell. A hyper matrix, i.e., a mathematical matrix of (matrices of (matrices… of (matrices or vectors or scalars))), corresponding to the anatomic region of interest is subsequently defined. The latter describes explicitly or implicitly the local biological, physical, and chemical dynamics of the region [2,3,14].

The basic mechanism of the model is based on the separation of biological cells within a geometric cell (otherwise known as volume element or voxel) through hypermatrix into equivalence classes. Each cell, depending on its mitotic potential and its current phase of the cell cycle, is assigned to one of these classes. Then, a series of status variables (oxygen and glucose concentration, cell number in the voxel and number of cells hit by therapy), and the application of transitional algorithms from a cellular state to another in periodical time intervals, determine the next overall state of the voxel and its cells.

The nephroblastoma Oncosimulator is implemented using the C++ language. The primary outputs of the model are a series of RAW image files displaying the tumor volume for each day of the treatment period and a set of DAT files containing all the numerical results (tumor growth percentage, etc.).

The nephroblastoma use case addresses directly two out of the four different categories of users defined in MyHealthAvatar, modelers (also corresponding to the more general category of biomedical or basic science researcher) and clinicians. Modelers are expected to focus mostly on the functionality of repositories, while their calls to the model execution module will generally be limited in number and their sole purpose would be to further calibrate and fine-tune their creations. On the other hand, doctors are expected to ask for a lot more model executions. As such, any involvement with the storage facilities and/or the API will be limited to uploading and retrieving medical data and saving result reports.

With the exception of medical data uploading, all other pertinent commands will be given through the tool execution module, to ensure minimal direct involvement with the repository CRUD mechanisms.

2.3.3. The breast cancer Oncosimulator

A second paradigm was based on another simulation model, the breast cancer Oncosimulator, which has been developed outside the scope of the MyHealthAvatar project. It aimed to demonstrate the versatility of the infrastructure by highlighting the general principles that were used to create the infrastructure database schema along with the workflows developed for the two different kinds of users and how they could be used to accommodate any kind of model.

The breast cancer Oncosimulator simulates the vascular tumor growth and the response to antiangiogenic treatment of breast cancer, through the administration of bevacizumab, a monoclonic antibody that prevents the connection of the vascular endothelial growth factor (VEGF) with the corresponding receptors on the endothelial cells surrounding the tumor. It is based on a system of ordinary differential equations, which describe the tumor volume and its carrying capacity, i.e. the maximum tumor volume that can be supported by the given vasculature [4,5]. The model is implemented as a set of MATLAB M files. One of them assumes the role of the master script and subsequently calls the others. The input parameters are fed as a Comma-Separated Values (CSV) file and the output is a diagram in which the tumor volume and its carrying capacity over time are plotted.

MATLAB is a language which, in contrast to lower level programming languages, such as C++, is very tightly coupled to its own execution environment, which is the MATLAB Compiler Runtime (MCR) [15]. This arrangement helped in demonstrating the ability of the infrastructure to address the end user's predicament of installing third-party software in their devices to access and use the tool and model repository contents. All stored models reside in the same physical machine as the infrastructure. In addition, the same device is used for model executions and result storage. Therefore, any and all software that is required for the model executions should be installed only in the infrastructure deployment machine. This allows access to the infrastructure using devices such as smartphones and tablets, which have less computational power in comparison to personal computers and laptops and therefore are less probable to have highly specialized software such as MCR installed within them.

2.3.4. Indicative Workflows

The modeler workflow is shown in Figure 2-3. After logging to the system, they enter the model repository. Assuming that the objective is to upload a simulation model for storage, data must be entered in at least three tables. First the general information table, where the name and description of the model will be given, followed by the uploading of all model pertinent files to the file table (executable file, documentation, auxiliary scripts for additional functions such as visualization of results). Finally, a list of input and output

parameters should be defined, which will facilitate the construction of the input form and the result report from the model execution module. Defining properties and relating them to the uploaded model is encouraged, albeit not compulsory.

Respectively, the clinician, after logging, would enter the repository only to register information of a new patient and/or to provide one or more sets of values for a certain patient, in order to use them subsequently as input for a simulation model. Then, they proceed to the model execution module forms. The interface wizard will prompt them to choose the following: model to use, and subject name, in that order. Then, a form appears with the appropriate input fields derived from the rows of the parameter storage table, which pertain to the chosen simulation model. After providing the desired inputs, the model performs an execution. The results were displayed on the last wizard screen and could be saved as a pdf file. Similarly, the clinician could download the saved report through the interface of this application. The workflow is shown in Figure 2-4.

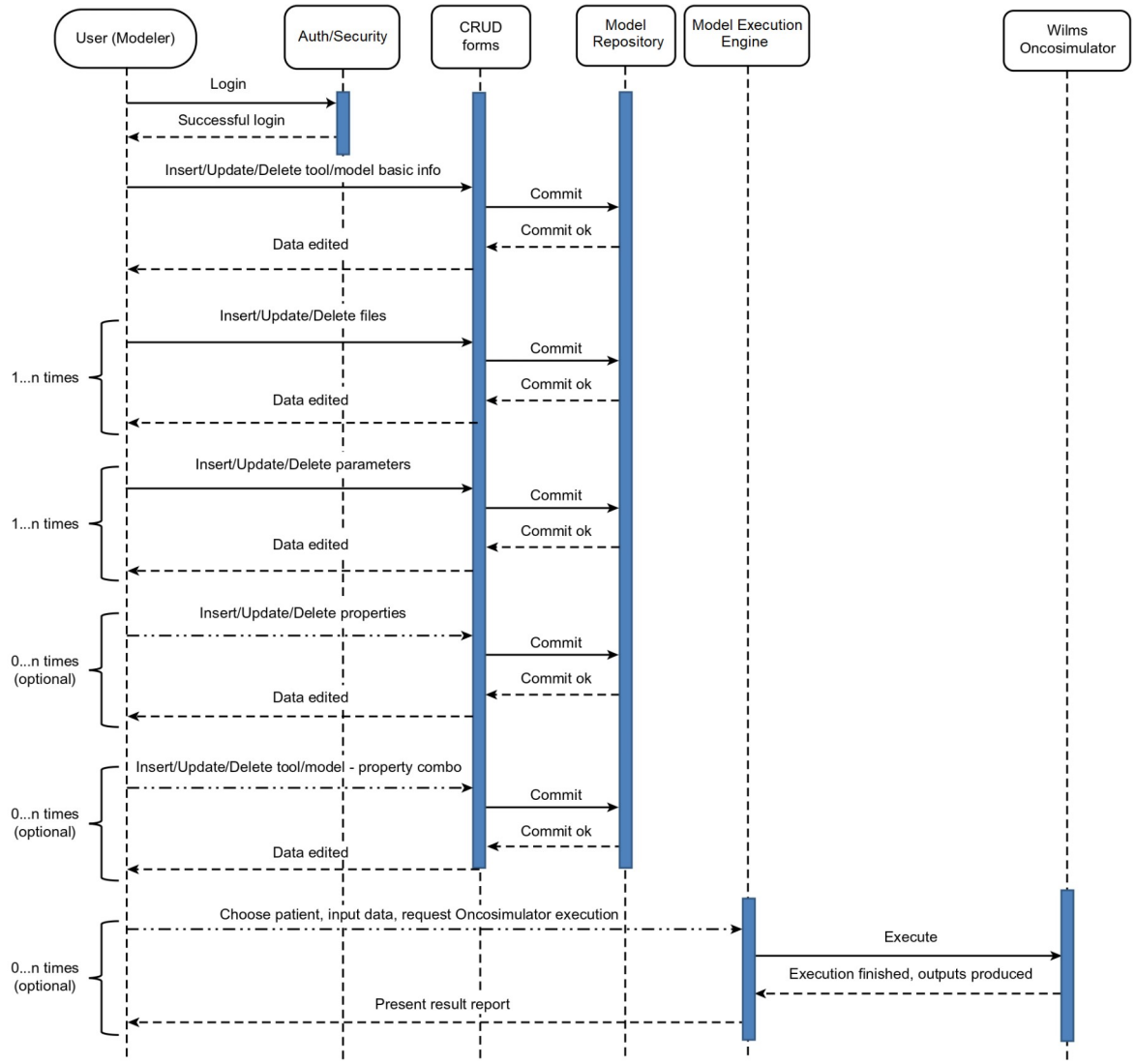


Figure 2-3. Modeler Workflow

2.3.5. Legal/ethics/security considerations

The development of the IAPETUS prototype as part of the MyHealthAvatar project was subject to a set of legal and ethical guidelines. These guidelines stemmed from the notion that the citizen becomes the main stakeholder and is able to freely upload and use their own data in conjunction with any and all available software tools developed in the project context. Especially in the case of patient-centered simulation models, privacy and data protection issues could arise. Prior to the use of data from a simulation model, the data owner should be aware of how the model will use their data, what new data will emerge from the simulation results and who will be the owner of these data. A simulation model should also operate under a security framework that prevents data loss or usage by unauthorized parties. Furthermore, the interpretation of execution results can lead to liability issues. A citizen is not necessarily expected to know the exact clinical meaning of all the data produced by a simulation model execution. Therefore, any attempts by the citizen to evaluate their own medical condition, outside of a clinical environment, could lead to misinterpretations and thereby negatively affect the person's psychological state. According to the doctrine of informed consent [16], the citizen must be informed about these issues before giving their consent for the use of their personal data.

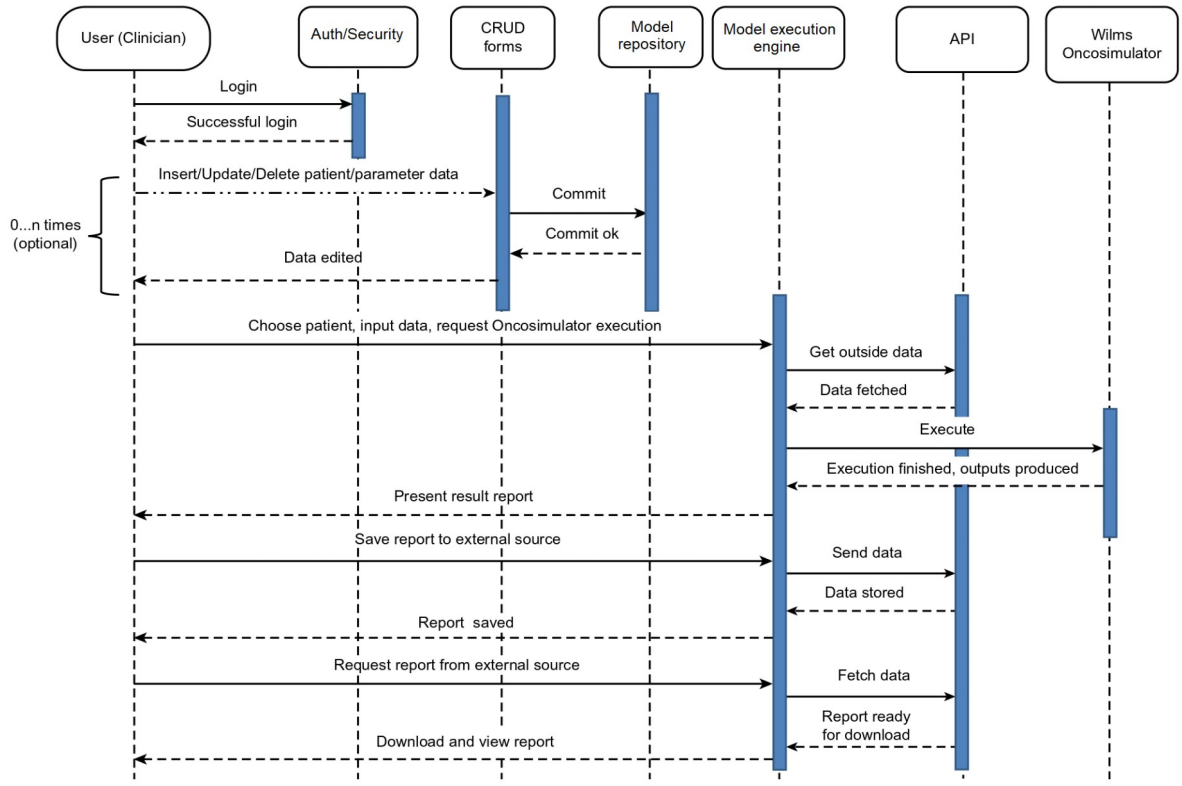


Figure 2-4. Clinician Workflow

The aforementioned issues, in conjunction with the types of stakeholders to whom the present work is addressed, dictated the choice and implementation of the IAPETUS security mechanisms. In order for a citizen's data to be used, a key assumption is made: a

clinician can use personal data, only under the specific informed consent of the data owner – in this case, the citizen – and on condition that both parties are interacting within the confines of a clinical environment. This means that the clinician and the citizen should be in the clinician's office, at the same time, in front of the same device (personal computer, laptop, tablet, etc.). The use of the citizen's data as input for one or more simulation models is allowed only if the citizen has previously given their permission for such actions to be taken and after an initial proper briefing regarding these actions has taken place. If the infrastructure is hosted in a large healthcare provider (e.g. a hospital), then a clinician can view, edit, and use the data that pertain only to the patients, whose consent they have already obtained. Similarly, to protect the modelers' intellectual property rights regarding the models they have created, a model and its related information (files, parameters, and property values) can be viewed by all IAPETUS users, but altered only by their creator. The creator can also determine which of the files that they uploaded to the infrastructure can be downloaded by other users and which files cannot.

In Django, all permissions are handled by the inherent authentication and authorization system. By explicitly specifying the user roles of modeler and clinician and defining the corresponding role rights, it is possible to prevent modelers from performing CRUD operations on personal patient data and clinicians from altering tool parts and specifications. Furthermore, the schema of the relational databases can be updated by placing additional fields to store the creator of an object, where necessary. Therefore, when a user views the object information, the comparison of the creator and viewer identities will enable or disable the editability of the returned template. In the case of machine-to-machine communication, the HTTPS protocol is used for the IAPETUS URL system, through the use of Python's Django-sslserver [17] package. Finally, if the users access the infrastructure from devices using static IP address settings, then Django offers the option to allow access only to specific IP addresses.

2.4. References

1. Spanakis E, Kafetzopoulos D, Yang P, Marias K. MyHealthAvatar: personalized and empowerment health services through internet of things technologies. *Wireless Mobile Communication and Healthcare (Mobihealth), 2014 EAI 4th International Conference on Wireless Mobile Communication and Healthcare – Transforming Healthcare Through Innovations in Mobile and Wireless Technologies*. Athens, Greece, 2014:331–4. doi:10.4108/icst.mobihealth.2014.257500.

2. Georgiadi E, Dionysiou D, Graf N, Stamatakos G. Towards in silico oncology: adapting a four dimensional nephroblastoma treatment model to a clinical trial case based on multi-method sensitivity analysis. *Comput Biol Med*. 2012;42(11):1064–78. doi: 10.1016/j.compbiomed.2012.08.008.

3. Stamatakos G, Georgiadi E, Graf N, Kolokotroni E, Dionysiou D. Exploiting clinical trial data drastically narrows the window of possible solutions to the problem of

clinical adaptation of a multiscale cancer model. *PLoS One*. 2011;6(3):e17594. doi: 10.1371/journal.pone.0017594.

4. Argyri KD, Dionysiou DD, Misichroni FD, Stamatakos GS. Numerical simulation of vascular tumor growth under antiangiogenic treatment: addressing the paradigm of single-agent bevacizumab therapy with the use of experimental data. *Biol Direct*. 2016;11(1):12. doi: 10.1186/s13062-016-0114-9.

5. Argyri KD, Dionysiou DD, Stamatakos GS. Modeling the interplay between pathological angiogenesis and solid tumor growth: the anti-angiogenic treatment effect. *Proceedings of the* 2012 5*th International Advanced Research Workshop* on In Silico Oncology and Cancer Investigation – The TUMOR Project Workshop *(IARWISOCI)*. Athens, Greece 22-23 Oct. 2012.

6. Stange KC, Zyzanski SJ, Jaén CR, et al. Illuminating the 'black box': a description of 4454 patient visits to 138 family physicians. *J Fam Pract*. 1998;46(5):377–89.

7. Hardt, D., Ed. (2012). *The OAuth 2.0 Authorization Framework* (RFC 6749). IETF.

8. The Web framework for perfectionists with deadlines | Django. *Djangoprojectcom*; 2016. Available at: https://www.djangoproject.com/. Accessed May 16, 2016.

9. Welcome to Tastypie! – Tastypie 0.13.3 documentation. *Django-tastypiereadthedocsio*; 2016. Available at: http://django-tastypie.readthedocs.io/en/latest/. Accessed May 16, 2016.

10. Homepage | Celery: Distributed Task Queue. *Celeryprojectorg*; 2016. Available at: http://www.celeryproject.org/. Accessed May 16, 2016.

11. RabbitMQ – Messaging that just works. *Rabbitmqcom*; 2016. Available at: https://www.rabbitmq.com/. Accessed May 16, 2016.

12. MongoDB for GIANT Ideas. *MongoDB*; 2016. Available at: https://www.mongodb.com/. Accessed May 16, 2016.

13. Models. Welcome to Django OAuth Toolkit documentation – Django OAuth Toolkit 0.10.0 documentation. Available at: https://django-oauth-toolkit.readthedocs.io/en/latest/. Accessed September 1, 2016.

14. Stamatakos G, Dionysiou D, Lunzer A, et al. The technologically integrated oncosimulator: combining multiscale cancer modeling with information technology

in the in silico oncology context. *IEEE J Biomed Health Inform*. 2014;18(3):840–54. doi: 10.1109/jbhi.2013.2284276.

15. MathWorks T. MATLAB Runtime – MATLAB compiler – MathWorks United Kingdom. Available at: http://www.mathworks.com/products/compiler/mcr/. Accessed September 1, 2016.

16. Forgó N, Kollek R, Arning M, Kruegel T, Petersen I. Ethical and legal requirements for transnational genetic research. *Eur J Int Law*. 2011;22(2): 614–7.

17. Django-sslserver 0.19: Python package index; 2016. Available at: https://pypi.python.org/pypi/django-sslserver/0.19. Accessed September 1, 2016.

3. Metadata Creation for In Silico Models – A theoretical approach

The increasing sophistication of computational models in biomedical research, particularly within the domain of in silico oncology, necessitates advanced strategies for managing the vast amounts of associated data. In addition, the emergence of hypermodels, which integrate diverse scales and facets of cancer, often comprising numerous interconnected sub-models representing biological processes at various levels. The data utilized and generated by these hypermodels are equally multifaceted, encompassing genomic information, imaging data, and clinical trial results [1]. Effective management of this intricate data, through well-defined metadata, is crucial for fostering collaboration, validating findings, and translating research into clinical practice [2].

As a part of the Computational Horizons In Cancer (CHIC) project [3] a theoretical infrastructure has been proposed, which leverages the principles of semantic web technologies to enhance the management of hypermodel metadata. Semantic infrastructure provides a framework that adds meaning and context to data, enabling more effective machine understanding and interoperability. This often involves the use of formal semantics to define relationships between data elements, creating a rich network of interconnected knowledge [4].

The proposed infrastructure emphasizes modularity, enabling its operation with diverse combinations of software tools, thereby offering flexibility and adaptability across various research environments [5]. The infrastructure is envisioned to connect with a broader metadata ecosystem to support content publishing, advanced search functionalities, and enhanced compliance with legal and ethical frameworks relevant to healthcare data [5]. Recognizing the distinct needs of different stakeholders, the design incorporates access rights for modelers to facilitate the upgrading and maintenance of their work, while providing end users with user-friendly interfaces for model execution and effortless retrieval of results [5]. The emphasis on a "semantic infrastructure layout" aligns with the growing recognition of the importance of semantic approaches for managing data and models in healthcare and life sciences [6].

3.1. Components Overview

As shown in Fig. 3-1, the overall system is comprised of seven main modules: the Initial Access Point, the Extract-Transform-Load (ETL) unit, two repositories (RDF data and knowledge base), and three front-end applications which provide users with access to the repositories and their data (annotation management, query management, knowledge base management).

3.1.1. User Roles

Individuals who will use the system will be given one of two sets of access rights. The Common User role will be used by people who want to only retrieve stored metadata by querying the RDF repository, such as citizens and clinicians, and consequently are allowed

to access only the querying application. The Special User role is reserved for personnel such as modelers and IT experts who update and maintain the information stored in the system. They have access to all of the front-end components and can thus modify the contents of both repositories.

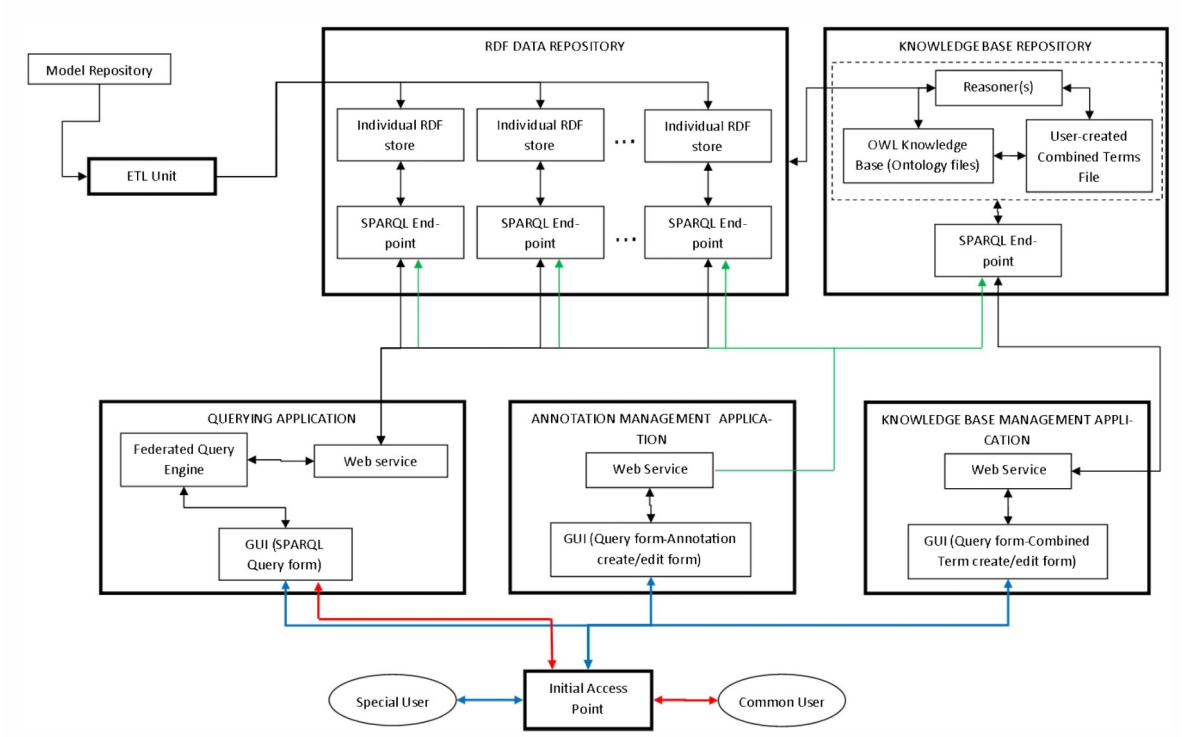


Figure 3-1. Schematic diagram of the proposed architecture layout

3.1.2. Entry Points

The system includes two "gateways" to communicate with the outside world. The ETL unit takes on the task of transforming the data contained in the Model Repository, into Resource Description Framework (RDF) statements, using mappings between the schema of the former and ontology terms. As described in chapter 2, this unit is included in the independent engines of the proposed hosting infrastructure layout. Depending on the implementation, the ETL unit may be either integrated into the repository or used as a separate application (standalone or a microservice) and called using REST APIs. Data can be entered in to the system either upon data entry in the model repository, or by setting up an event (cronJob, etc.) and allow the ETL mechanism to run periodically and update the data.

The Initial Access Point is the first module encountered by anyone who wants to log into the system. It identifies the users who request access and enforces the user roles by granting access to the proper modules.

3.1.3. User Communication Components

The front end of the system consists of three applications, which are responsible for user interaction with the system's repositories. Access to the contents of the former is

provided through user query submissions. Furthermore, only special users are allowed to insert, edit, or delete data. These applications connect remotely to the repositories through web services which communicate with the formers' available SPARQL endpoints.

The **annotation management application** allows a special user to create mappings between model resources and terms from ontologies contained in the knowledge base repository. After entering the application, the user is provided with the ability to complete the "annotation create / edit" form with the IDs of a resource and an ontology term in order to create the desired mapping. Alternatively, they can submit queries to the appropriate form for the retrieval of any of the two elements. For this reason, the application is required to be connected with both repositories. Finally, a query can be submitted for the retrieval of an existing annotation so that the user can edit it. It should be noted at this point that this application is connected to only one of the RDF repository's SPARQL endpoints at any given session and can use data and annotations published to that specific endpoint. Since the RDF repository is intended to hold data from different sources (hospitals, health organizations, research institutes, etc.) separately, each user must state their employer upon registering to the system. As a result, any user actions will affect only their employer's respective endpoint. Should a person have multiple employers, they must still choose one of them when they sign in, in order for their work to affect only one endpoint.

The **knowledge base management application** has similar basic components to those of the previous application. However, this application communicates only with the knowledge base repository in order to combine terms from the different ontologies stored there, using logical operations, in order to represent complex biological concepts. These terms can later be used by the annotation management application to effectively annotate these concepts. A user can either directly use the create/edit form to input ontology term IDs, the desired operations, and create a combined term, search for said IDs if needed via the query form, or retrieve a combined term in order to modify or delete it. The new terms are stored in a separate RDF file, which resides at the knowledge base repository and also contains links to the stored ontologies. This prevents multi-recording a combined term ID in each of the ontology files, the terms of which it is made from.

The **query application** is the most crucial part of the front end, as it is responsible for the communication between the users and the RDF repository. It is accessible by all individuals regardless of their role. It includes a query form, in which the user submits the query to the repository in SPARQL language. In this module, the system's federated query engine is included. This tool receives the input query from the query form and breaks it into individual subqueries in order to submit them to the possibly more than one different SPARQL endpoints which publish the repository's data. This provides transparent access to the contents of the latter, since the non-use of SERVICE and BINDINGS clauses in the original query's body means that no prior knowledge of the data origin (which information is published by which endpoint) or how they will be retrieved is required by the end users.

3.1.4. Storage Components

This is the backbone of the entire system, which is practically divided into two parts due to the partially different kinds of data stored (changeable totals RDF statements as opposed to OWL ontologies) and the different access regulations that apply to each part which result from the number and type of users that request to log in.

The **RDF repository** stores the bulk of the metadata in the form of RDF statements. These metadata are derived from the operation of the ETL unit and the annotation management application and can reach millions of statements in number. It also provides the ability to create SPARQL endpoints, through which user-submitted remote queries are handled and their responses are returned. This repository is expected to be used by a number of different institutions which handle data of the scientific field. The basic requirement is to make these data available to the public. At the same time, however, each institution seeks to independently maintain control of their own information and bear responsibility for keeping them up to date. A proposed solution is a virtual "partition" of the repository and assigning each part (otherwise called an individual RDF store for convenience) to a separate SPARQL endpoint. That way, each stakeholder can seamlessly perform any desired changes. This solution dictates the use of the federated query engine, so that the end user is given the impression that there is only one repository.

The **knowledge base repository** is smaller in size than the RDF repository and is accessible only by special users. It contains the ontologies, the terms of which are used for the annotation of resources, the file that contains the combined terms created by the knowledge base management application, and one or more semantic reasoners, which are used to produce new statements based on the existence of others, which are regarded as axioms. In addition to any known reasoners (Fact++ [7], Pellet [8], etc.) which are suitable for OWL ontologies, additional rule files may be stored, which extend the former and are based on the specific characteristics of the system. The reasoners can also be available to the RDF repository to allow the application of their rules directly on the RDF data or analyze the SPARQL queries that posed to them, so the answer contains all the necessary additional statements, or by applying incremental reasoning [9]. Finally, the existence of a SPARQL endpoint serves as the module's communicator with both front-end applications with which it cooperates.

3.2. The CHIC Project Paradigm

The aim of the project "Computational Horizons In Cancer (CHIC): Developing Meta- and Hyper-Multiscale Models and Repositories for In Silico Oncology" was to address the complexity of cancer and to describe the phenomena that are being caused by it at the various biological levels of the human body (molecular, cellular, tissue, organ, etc.). As the representation of the entire disease with a single model is not possible, hypermodels are used; hypermodels are models made of elementary component models, or otherwise called, hypomodels. This leads to the creation of hyper-terms which must be annotated,

while the resources that hypomodels can be annotated with are expected to be combined in the same way as the latter, thus resulting in the formation of hyper-resources[3].

For that reason, the project included a task whose objective was to create an infrastructure capable of coping with the management of all this metadata. The heart of this infrastructure was an evolved form of the solution developed in the context of the RICORDO project [10]. The goal was to extend the provided infrastructure in order to meet the needs of CHIC. This proposal seeks to achieve this goal through the decomposition of the given RICORDO infrastructure. This is done in order to enrich it with any additional required software components or to upgrade some of the existing ones. In this case, the example of the hypermodeling creation process, the research for which is ongoing, is followed.

A pertinent investigation was carried out, giving special attention to the RDF repository and the federated query machine. In order to fully exploit the project's private cloud, distributed open-source solutions are considered for the repository, such as HDRS [11], 4store [12] and Virtuoso server's [13] clustered edition. For the federated query engine, free, open-source software packages such as SPARQL-DQP [14], ANAPSID [15], FedX [16], and ADERIS [17] are considered. Each of them implements a different approach to achieve SPARQL endpoint federations, and the capabilities for contribution in the overall result are being explored. Consequently, any changes might affect other components, such as the knowledge base, having being implemented by a combination of OWLlink server [18] with the Pellet reasoner.

3.3. Suggested expansions

Building upon the foundation laid out in the initial proposal, several key areas for future research and development emerge. One crucial direction involves a deeper investigation into the scalability and performance of the proposed semantic infrastructure when dealing with the large datasets characteristic of in silico oncology research [19]. Exploring advancements in linked data technologies, the increasing availability of comprehensive biomedical ontologies [20], and the development of more efficient semantic data stores [21] could lead to significant enhancements. Another promising avenue lies in the integration of advancements in artificial intelligence (AI) and machine learning (ML) [22] to automate metadata extraction and semantic annotation processes, thereby reducing manual effort and improving efficiency. Furthermore, exploring the potential for integrating the infrastructure with existing in silico oncology platforms and linked open data initiatives in the biomedical domain [23] could foster broader adoption and interoperability. The development of user-friendly interfaces and application programming interfaces (APIs) would also be critical for facilitating wider access and utilization by the research community.

3.4. References

1. Kolokotroni, E.; Abler, D.; Ghosh, A.; Tzamali, E.; Grogan, J.; Georgiadi, E.; Büchler, P.; Radhakrishnan, R.; Byrne, H.; Sakkalis, V.; Nikiforaki, K.; Karatzanis,

I.; McFarlane, N.J.B.; Kaba, D.; Dong, F.; Bohle, R.M.; Meese, E.; Graf, N.; Stamatakos, G.; on behalf of the CHIC Project Consortium. A Multidisciplinary Hyper-Modeling Scheme in Personalized In Silico Oncology: Coupling Cell Kinetics with Metabolism, Signaling Networks, and Biomechanics as Plug-in Component Models of a Cancer Digital Twin. J. Pers. Med. 2024, 14, 475.

2. Ulrich H, Kock-Schoppenhauer AK, Deppenwiese N, Gött R, Kern J, Lablans M, Majeed RW, Stöhr MR, Stausberg J, Varghese J, Dugas M, Ingenerf J. Understanding the Nature of Metadata: Systematic Review. J Med Internet Res. 2022 Jan 11;24(1):e25440. doi: 10.2196/25440. PMID: 35014967; PMCID: PMC8790684.

3. Stamatakos G, Dionysiou D, Misichroni F, Graf N, van Gool S, Bohle R, Dong F, Viceconti M, Marias K, Sakkalis V, Forgo N, Radhakrishnan R, Byrne H, Guiot C, Buechler P, Neri E, Bucur A, de Bono B, Testi D, Tsiknakis M. Computational Horizons In Cancer (CHIC): Developing Meta- and Hyper-Multiscale Models and Repositories for *In Silico* Oncology - a Brief Technical Outline of the Project. Proc 2014 6th Int Adv Res Workshop In Silico Oncol Cancer Investig (2014). 2014 Nov;2014:10.1109/iarwisoci.2014.7034630. doi: 10.1109/iarwisoci.2014.7034630. PMID: 34541587; PMCID: PMC8444280.

4. Semantic Technology and Integration 101: What It Is and Why It Matters - Dataversity, accessed on April 19, 2025, https://www.dataversity.net/semantic-technology-and-integration-101-what-it-is-and-why-it-matters/

5. N. A. Christodoulou and G. S. Stamatakos, "A modular semantic infrastructure layout for the management of hypermodel-pertinent metadata in the context of In Silico oncology," *Proceedings of the 2014 6th International Advanced Research Workshop on In Silico Oncology and Cancer Investigation - The CHIC Project Workshop (IARWISOCI)*, Athens, Greece, 2014, pp. 1-4, doi: 10.1109/IARWISOCI.2014.7034640.

6. Haque AKMB, Arifuzzaman BM, Siddik SAN, Kalam A, Shahjahan TS, Saleena TS, Alam M, Islam MR, Ahmmed F, Hossain MJ. Semantic Web in Healthcare: A Systematic Literature Review of Application, Research Gap, and Future Research Avenues. Int J Clin Pract. 2022 Oct 18;2022:6807484. doi: 10.1155/2022/6807484. PMID: 36320897; PMCID: PMC9596238.

7. D. Tsarkov and 1. Horrocks , "FaCT++ Description Logic Reasoner: System Description," in Automated Reasoning, vol. 4130, U. Furbach, N. Shankar, Ed. Berlin Heidelberg: Springer, 2006, pp. 292-297.

8. E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur and Y. Katz, "Pellet: A practical OWL-DL reasoner," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 5, no.2, Jun 2007, pp. 51-53.

9. B. Parsia, C. Halaschek-Wiener, and E. Sirin, "Towards incremental reasoning through updates in OWL-DL," presented at the 2006 151h Int. World Wide Web Conf; Edinburgh, Scotland.

10. S. M. Wimalaratne, P. Grenon, R. Hoehndor(G. V. Gkoutos and B. de Bono, "An infrastructure for ontology-based information systems inbiomedicine: RICORDO case study," Bioinformatics, vol. 28, no. 3,Nov 20 I I, pp. 448-450.

11. C. Bohm, D. Hefenbrock, and F. Naumann, "Scalable peer-to-peer-based RDF management," in Proc. 8th Int, Conf on Semantic Systems, New York, 2012, pp. 165-168.

12. Harris S, Lamb N, Shadbolt N. 4store: The Design and Implementation of a Clustered RDF Store. In: *The Semantic Web - ISWC 2009*. Springer Berlin Heidelberg; 2009:81-96. doi:10.1007/978-3-642-04930-9_6

13. O. Erling and 1. Mikhailov, "RDF Support in the Virtuoso DBMS," in Networked Knowledge - Network Media, vol. 221, T. Pellegrini, S. Auer, K. Tochtermann and S. Schaffert, Ed. Berlin Heidelberg: Springer, 2009, pp. 7-24.

14. C. Buil-Aranda, M.Arenas and O.Corcho, "Semantics and Optimization of the SPARQL 1.1 Federation Extension,", in The Semantic Web: Research and Applications, vol. 6644, G. Antoniou, M.Grobelnik, E. Simperl, B. Parsia, D. Plexousakis, P. De Leenheer and 1.Pan, Ed. Berlin Heidelberg: Springer, 20 I I, pp. 1-15.

15. M. Acosta, M. Vidal, T. Lampo, 1. Castillo and E. Ruckhaus, "ANAPSTD: An Adaptive Query Processing Engine for SPARQL Endpoints," in The Semantic Web - ISWC 201I, vol. 7031, L. Aroyo, C. Welty, H. Alani, 1. Taylor, A. Bernstein, L. Kagal, N. Noy and E. Blomqvist, Ed. Berlin Heidelberg: Springer, 2011, pp. 18-34.

16. A. Schwarte, P. Haase, K. Hose, R. Schenkel and M. Schmidt, "FedX: Optimization Techniques for Federated Query Processing on Linked Data," in The Semantic Web - ISWC 2011, vol. 7031, L. Aroyo, C. Welty, H. Alani, 1. Taylor, A. Bernstein, L. Kagal, N. Noy and E. Blomqvist, Ed. Berlin Heidelberg: Springer, 2011, pp. 601-616.

17. Lynden, S., Kojima, I., Matono, A., Tanimura, Y. (2011). ADERIS: An Adaptive Query Processor for Joining Federated SPARQL Endpoints. In: Meersman, R., *et al.* On the Move to Meaningful Internet Systems: OTM 2011. OTM 2011. Lecture Notes in Computer Science, vol 7045. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25106-1_28

18. Thorsten Liebig, Marko Luther, Olaf Noppens, Mariano Rodriguez, Diego Calvanese, Michael Wessel, Matthew Horridge, Sean Bechhofer, Dmitry Tsarkov, and Evren Sirin. OWLlink: DIG for OWL 2. *OWLED 2008, 5th OWL Experienced and Directions Workshop, Karlsruhe, Germany, October 2008*.

19. Bispo KA, Rosa NS, Cunha PRF. SITRUS: Semantic Infrastructure for Wireless Sensor Networks. *Sensors*. 2015; 15(11) :27436-27469. https://doi.org/10.3390/s151127436

20. Knowledge Representation and Semantics - Working Group, accessed on April 19, 2025, https://amia.org/community/working-groups/knowledge-representation-and-semantics

21. MULTISENSOR Deliverable 5.1: Basic semantic infrastructure - CORDIS, accessed on April 19, 2025, https://cordis.europa.eu/docs/projects/cnect/1/610411/080/deliverables/001-D51BasicSemanticInfrastructure20140630v11Ares20143836635.pdf

22. Yan, S. (2023). Research on Metadata in the Era of Big Data Based on Bibliometric Analysis by CiteSpace. In *Proceedings of the 2022 3rd International Conference on Big Data and Informatization Education (ICBDIE 2022)* (pp. 246–255). Atlantis Press International BV. https://doi.org/10.2991/978-94-6463-034-3_25

23. N.F. Noy Semantic integration: a survey of ontology-based approaches. SIGMOD Rec., 33 (4) (2004), pp. 65-70, https://doi.org/10.1145/1041410.1041421

4. Integration with Clinical Decision Support Systems (CDSS)

The growing realization that cancer is a complex and heterogeneous disease has shifted oncology to the notion of personalized medicine, which advocates the tailoring of treatment strategies to the individual characteristics of each patient so that therapeutic effect can be maximized, side effects are reduced and patient overall life quality is improved [1]. Towards such a paradigm has included, Clinical Decision Support Systems (CDSS) have now emerged as critical instruments meant to assist clinicians in making very well-informed data-driven decisions at the point of care [3]. Adding up to this development are advancing multiscale modeling approaches of cancer such as the Oncosimulator, which can mimic tumor growth and drug response on an individual basis [2]. This chapter will explore the integration of the Oncosimulator paradigm with CDSS to facilitate personalized medicine, focusing on its application in the context of nephroblastoma and breast cancer.

4.1. CDSS definition

A Clinical Decision Support System (CDSS) is defined as any system specifically designed to enhance clinical decision-making processes related to diagnosis or treatment [3]. Initially conceived over four decades ago, CDSS have undergone significant evolution, transitioning from simple rule-based systems to increasingly sophisticated platforms that now incorporate elements of artificial intelligence and machine learning [3]. This progression reflects the advancements in computing power and the growing capacity to analyze complex medical datasets, enabling CDSS to provide more nuanced and relevant support to clinicians [3]. Modern CDSS are frequently integrated into Computerized Provider Order Entry (CPOE) systems and Electronic Health Records (EHRs), aiming to prevent prescribing errors and other types of medical errors by offering features such as default values for drug doses, routes of administration, and frequency, as well as checks for drug allergies and interactions [3].

The architecture of a typical CDSS comprises three fundamental components [6]. First, the knowledge base serves as a repository of medical information, encompassing clinical guidelines, protocols, and best practices. This knowledge is often structured in the form of IF-THEN rules or represented through machine learning models [6]. Second, the inference engine is the core processing unit that applies the rules or algorithms from the knowledge base to the specific data of a patient, thereby generating recommendations or alerts [6]. Finally, the user interface is the mechanism through which the CDSS presents its findings to the user, which can be a clinician, a patient, or another healthcare professional. This interface can manifest as dashboards, alerts, pop-up messages, or modules seamlessly integrated within EHR systems [6].

CDSS supports many features related to the different aspects of the practice of medicine [3]. Alerts and reminders can thus be created for such critical items as doses of medication, drug interactions, known allergies, and preventive measures [3]. Diagnostic support is offered through the analysis of patient data and arriving at possible diagnoses [3]. Furthermore, treatment recommendations can be made based on clinical guidelines and the individual specifics of each patient [3]. Functionality extends to order facilitators that help

choose appropriate tests, medications, and procedures, as well as display relevant and updated patient information to the clinicians. In addition to supporting direct clinical work, CDSS may also play a role in optimizing clinical workflows, helping with risk prediction and prevention, and even patient engagement and education through personalized health information [7].

CDSS can be categorized into several types based on the mechanisms behind each type and their applications [6]. Knowledge-based CDSS depend on a rule and relation based structure for storing knowledge derived from medical data; it is most frequently in the form of IF-THEN logic, to provide advice about clinical activities. On the other hand, non-knowledge-based CDSS rely on artificial intelligence, particularly machine learning, to train with data generated historically, in order to detect any occurrence of patterns. Examples of non-knowledge-based systems include those utilizing artificial neural networks (ANN) and genetic algorithms (GA) [6]. CDSS can be further categorized by the timing of their use in the clinical process, with respect to prediagnostic systems that aid a diagnosis preparation; during-diagnosis systems which help in reviewing and filtering diagnostic options; and postdiagnosis systems for data mining and predicting the events. Additionally, the classification may include whether the CDSS is passive in that it provides information for clinicians as they see fit, or active because it interacts directly with users through alerts and recommendations integrated into the clinical workflow [12].

4.2. How CDSS Workflows Can Help Personalized Medicine

The role of workflows within CDSS is critical to personalized medicine, for they permit integration and analysis of a multitude of patient-specific data [1]. These various workflows can be designed to incorporate information from diverse sources such as EHRs that provide detailed patient histories and accounts of clinical encounters, laboratory results that reflect physiological states, imaging studies that give insight into various anatomical and pathological features, genomic profiles that proffer a molecular interpretive framework for diseases, and patient-generated health data that capture real-world experiences and outcomes [1]. An analysis of this broad set of information can give the clinician a much deeper understanding of the unique condition of each patient and their likely response to a variety of treatment approaches [1].

A particularly significant aspect of personalized treatment in oncology is the role of integration of genetic and molecular information into treatment decisions. The CDSS workflows can be customized to facilitate processing and interpretation of the large amounts of genetic and molecular data, which have increasingly become cornerstones of cancer biology and have been accepted as predictors of successful response to anticancer treatment [13]. Therefore, target therapies can be chosen that are designed to work against the specific molecular alterations that facilitate an individual patient's tumor growth and progression [1]. This type of approach is called precision oncology, and it is supposed to deliver more effective therapies with fewer side effects than traditional concepts of therapy such as chemotherapy or radiotherapy.

By utilizing the detailed analysis of patient-specific information and including relevant medical knowledge, CDSS workflows can suggest treatment recommendations that are exquisitely tailored to a patient's individual needs and preferences [1]. This represents a radical departure from the one-size-fits-all approach that has long characterized cancer medicine [14]. Considering the unique biological, clinical, and even personal factors of each patient, CDSS can guide clinicians toward the most appropriate therapeutic strategy, ensuring maximum chances of success in a high number of cases.

Moreover, CDSS workflows can empower patients towards active involvement in their care by sharing personalized health information and rendering precise treatment options to them [7]. This aspect is in accordance with the principles of patient-centered care that are fundamental to personalized medicine [1]. By facilitating shared decision-making, CDSS could ensure that treatment plans are both biologically sound and aligned with a patient's values, preferences, and overall goals for health and well-being.

4.3. The Oncosimulator Paradigm: A Foundation for Personalized Oncology

The Oncosimulator paradigm represents a sophisticated approach to multiscale cancer modeling, embodying the core principles of personalized oncology [2]. At its core, the Oncosimulator is a concept that encompasses multilevel integrative cancer biology, a complex algorithmic construct, and a powerful software tool [2]. Its primary objective is to assist the clinician in refining the treatment of cancer for a particular patient by simulating how that particular patient's tumor will respond to various treatment schemes in vivo [2]. The Oncosimulator acts not only for optimization procedures in treatment but is also a platform to develop insights into the core biological processes of cancer, support the design and interpretation of clinical trials, and serve as an educational tool for doctors, researchers, and even interested patients [2]. Ultimately, the Oncosimulator intends to develop a "digital twin" of the patient's tumor, an in silico representation for the prediction of treatment outcomes before the actual therapies are executed [8].

Using a highly sophisticated multiscale modeling technique, the Oncosimulator has considered biological phenomena across levels of complexity-from the molecular and cellular to that of tissues, organ, and entire body system [2]. It often utilizes a "top-down" simulation strategy, which begins with macroscopic imaging data obtained from the patient and then progressively incorporates details from lower levels of biological organization [2]. This approach is frequently implemented using discrete entity—discrete event cancer simulation techniques, which model the behavior of individual cells and their interactions over time [2].

For the Oncosimulator to achieve such objectives requires the integration of various key features and technologies since it is capable of integrating multiscale data from multiple sources [2]. Tumor segmentation and characterization are achieved through the use of image processing techniques that analyze medical imaging data, including MRI and PET scans [2]. To define an anatomical region of interest and its complex biological dynamics, a mathematical construct known as hypermatrix is often employed [2]. At the cellular level,

Oncosimulator usually takes a generic cytokinetic model of tumor cells describing basic processes including cell cycling, entry and exit from dormancy, differentiation into various cell types, and cell death mechanisms targeted explicitly by the cytotoxic effects of chemo- and radiotherapy [1]. As these simulations are computationally intensive, the Oncosimulator can utilize grid computing infrastructures to accelerate their execution in high-performance manner [2]. Predictions from simulations are visualized by various means, including 2-dimensional graphs, 3-dimensional rendering showing size and shape of tumor, and even 4-dimensional visualization illustrating the evolution of the tumor with time [2]. Finally, a Data Management System (DMS) plays a significant role in the management and storage of large quantities of patient-specific data necessary for the simulations [2].

4.3.1. Integrating the Oncosimulator with Clinical Decision Support Systems

For the Oncosimulator to be effectively utilized for clinical practice, it must be integrated within existing clinical workflow systems through Clinical Decision Support Systems (CDSS) [4]. This should make timely and relevant information derived from the simulations readily available to clinicians at the point of care to support support their decision-making. Various modalities of integration may be adopted, including making the Oncosimulator accessible through a web service that could be accessed by a CDSS, or embedding the functions of Oncosimulator directly into the architecture of a CDSS framework [2]. Particularly important in this integration is ensuring robust data exchange and interoperability between the Oncosimulator and the CDSS, thus making them able to communicate properly and share sufficient patient information and simulation results [9].

The Oncosimulator functionalities should therefore naturally fit within the workflow of CDSS for successful integration [12]. This also involves considering how a clinician will use the integrated system from the very first step of patient data entry to the last step of receiving and interpreting the simulation results [7]. Challenges during implementation should also be noted, such as delegating extensive patient data entry into the system to the clinicians, risk of alert fatigue if the system generates too many notifications, and the need for comprehensive training to ensure that users can effectively utilize the integrated system.

Seamless data exchange and robust interoperability are crucial for the effective integration of the Oncosimulator with CDSS [4]. This means carefully considering what types of data should be shared between both systems, which usually entails patient demographics, detailed medical history, imaging data in varying formats and treatment specification parameters [2]. It is through standardized formats for data and well-defined ontologies that the exchange is enabled, so both systems understand and utilize the information being shared [9].

4.3.1.1. The breast cancer Oncosimulator

The objective of the breast cancer Oncosimulator is to simulate the response of clinical breast tumors to specific treatment schemes and/or schedules in the patient individualized context. To this end, a continuum approach describing vascular tumor growth

under angiogenic signaling has been developed based on relevant literature [10]. and bevacizumab pharmacokinetic properties [11]. bj676

All major biological phenomena of cancer cell population dynamics are incorporated into the model, i.e. cancer cell proliferation, cancer cell apoptosis, post-vascular dormancy (state where tumor growth ceases due to the balance achieved between proangiogenic and antiangiogenic factors), endothelial cell death, spontaneous loss of functional vasculature, excretion of endogenous proangiogenic factors (such as vascular endothelial growth factor, fibroblast growth factors, platelet-derived growth factor, angiopoietin-1 etc.), excretion of endogenous anti-angiogenic factors (angiostatin, endostatin, angiopoietin-2 etc.) and antiangiogenic treatment – induced endothelial cell death as well as the resulting cancer cell death.

The implicit assumptions on which the basic framework of the model is based are that the tumor is a three dimensional spheroid, the diffusion process is in a quasi-stationary state, i.e. the tumor growth rate as well as the rate of change of drug concentration are relatively small compared to the rate of distribution of angiogenesis stimulators, and the concentration of the stimulator is a radially symmetric function.

The clinical questions have been chosen to be represented in the CDSS are the following [4]:

1. **Given the treatment scheme to be administered what is the outcome of a user-specified intermittent bevacizumab monotherapy to a specific breast tumor at the end of the treatment?**
   In this scenario the clinician can either select between two suggested modes of administration (dosage and frequency of infusions in accordance with European Medicines Agency) or experiment with all drug administration details in order to select the optimal treatment scheme for a specific patient from a set of candidate ones. Optionally, one can also request a desired percentage of tumor reduction. In this case, the system gives an extra response as to whether the clinical target is accomplished at the end of the treatment or not.
2. **How is the treatment outcome affected by applying fractionated versions of an original treatment scheme?**
   In this scenario the clinician can either select between two suggested modes of administration (dosage and frequency of infusions in accordance with European Medicines Agency). Alternatively, the clinician can experiment with the frequency of infusions or with the administered dosage in order to create a fractionated (spread out over time) scheme. In the latter case, the dosage or the frequency of infusions, respectively, are adjusted in an automated way. Hence, the clinician may select among a continuous administration of a given amount of drug spread during the whole therapeutic cycle or a typical optimal biological dose scheme where the dose is given as a few concentrated administrations or any intermediate version of the original treatment scheme.

4.3.1.2. The Wilms tumor Oncosimulator

The Wilms tumor (Nephroblastoma) Oncosimulator is an integrated software system simulating the growth of nephroblastoma tumors and their in vivo response to chemotherapeutic modalities within the clinical trials environment aiming to support clinical decision making in individual patients. The Nephroblastoma Oncosimulator has been developed by the In Silico Oncology and In Silico Medicine Group of the Institute of Communication and Computer Systems, National Technical University of Athens within the framework of P-Medicine project and a European clinical trial SIOP (International Society of Paediatric Oncology). The scenario which will demonstrate the use of Wilms Oncosimulator by the Clinical Decision Support System simulates a protocol of preoperative chemotherapy with a combination of actinomycin-D and vincristine for unilateral stage I-III nephroblastoma tumors. The clinical question to be addressed is the preferable chemotherapeutic scheme in terms of the administration points of the chemotherapeutic drugs. The goal is to detect the optimal combination of day points in which the 4 vincristine and 2 actinomycin-D dosages should be administered. In this context, the provision of a clinical decision support tool might prove of particular importance

4.3.2. Integration workflow and results

The integration of the Oncosimulator models is based on publishing models as dedicated engines, via a client application, written in Java and designed to connect to the server application comprising the core of the CDS. As the two models were implemented in different programming languages, the models were incorporated in a wrapper to achieve the required integration and build the services; for each Oncosimulator model a different procedure of transformation and integration was followed (Fig. 4-1); the breast cancer Oncosimulator is developed in MATLAB, as a master script, calling a number of functions; therefore, by utilizing the MATLAB Compiler, the model was recompiled into an executable java jar, with its main runner class being mapped to the master script; the jar was finally included in the wrapper's Maven repository; the nephroblastoma branch, written in C++, was recompiled in a form of shared library using the Java Native Interface [5], by changing its main function into a function which communicates with the wrapper code through a header file; both procedures produce corresponding wsdl files, which describe the web services [4].

To demonstrate the integration of the Oncosimulator into the CDS system, the scenarios based on the clinical questions stated in the previous sections were considered. Under the standard CDSS workflow, the clinician selects the corresponding model and sets values to user generated input parameters (treatment schema). By pressing "run", a request is sent to the client service using the URL of the remote machine where the model was published, together with the necessary input data. These data are consolidated into a file (csv for breast cancer, xml for nephroblastoma), which serves as input to the model. After the execution is completed, the output files are saved into the remote machine and a predefined set of results depicting clinically important information (the required set of

biological values, along with a graph depicting the tumor evolution over time), are sent as response and shown in the CDSS GUI (Fig. 4-3) [4].

The scenario is presented via an exemplar case study based on a model instance already developed using time-course data derived in the context of in vivo experiments studying the anti-tumor efficacy of bevacizumab in higher mammalian species (Mus musculus); the way that the clinician interacts with the system is sufficiently generic and as such will not be affected by the substitution of this model instance with another referring to human patient cases and which is currently under refinement; the details of both the original and the fractionated version of the scheme appear in Fig. 4-2 [4].

Even though the treatment schemes did not induce tumor regression for the specific tumor, tumor growth inhibition (i.e. comparing to the tumor evolution in the case of untreated growth) has been achieved for both cases; in particular, as it is shown in Fig. 4-3,

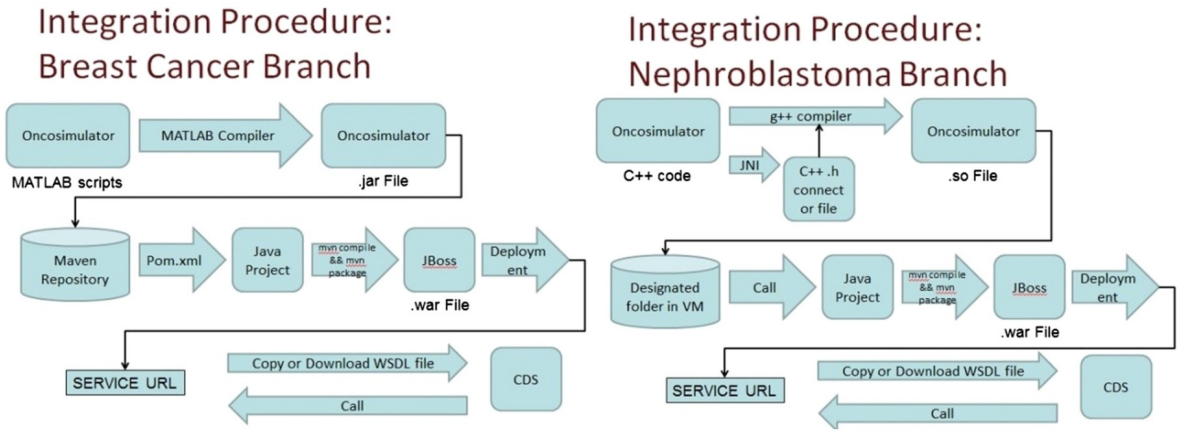


Figure 4-1: Integration procedure of breast cancer and nephroblastoma Oncosimulators

the original treatment scheme (blue line) has induced a tumor growth inhibition percentage equal to 73.6 % while the fractionated one (green line) has prompted a tumor growth inhibition percentage equal to 79.5 % (calculated as the percentage of change in tumor volume in the treatment module simulation with respect to the free growth module simulation, at the end of the treatment cycle); hence, the superiority of the fractionated scheme comparing to the original scheme would be revealed to the user along with the implied statement that none of the applied modes of administration would induce tumor regression [4].

For the Wilms Tumor Oncosimulator the impact which would have a delay of one dose of the preoperative scheme defined by the SIOP 2001/GPOH clinical trial protocol for unilateral stage I-III nephroblastoma tumors is studied. The study of the effect of two days delay (e.g. due to weekend) in the second administration of vincristine and shifting or not of the rest of the treatment scheme is presented. A real patient case from SIOP 2001/GPOH clinical trial is used. It was assumed that the chemotherapy starts one day after diagnosis and that surgical removal of the tumor is made 25 days after diagnosis. The two treatment schemes simulated are presented in table 4-1. The results are presented in table 4-2 and figure 4-4 [4].

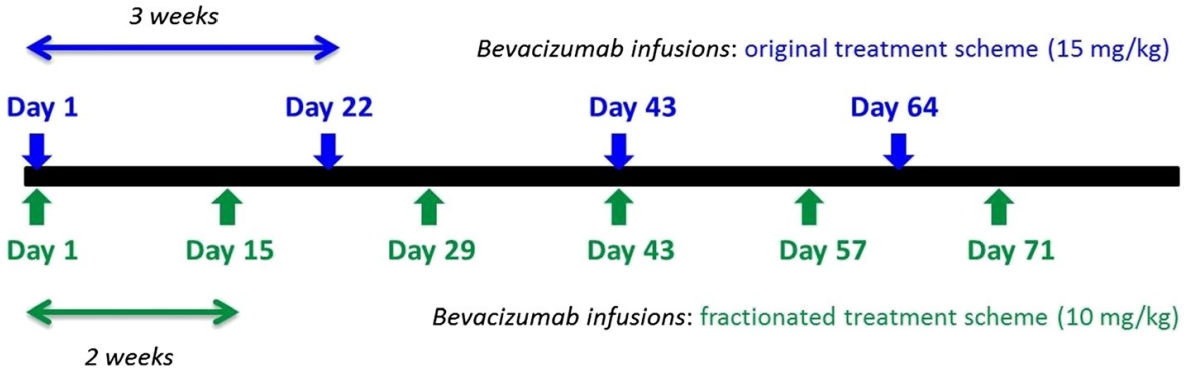


Figure 4-2: Breast cancer Oncosimulator: details of the two simulated bevacizumab monotherapy schemes (dosage, frequency of administration and time points of administration).

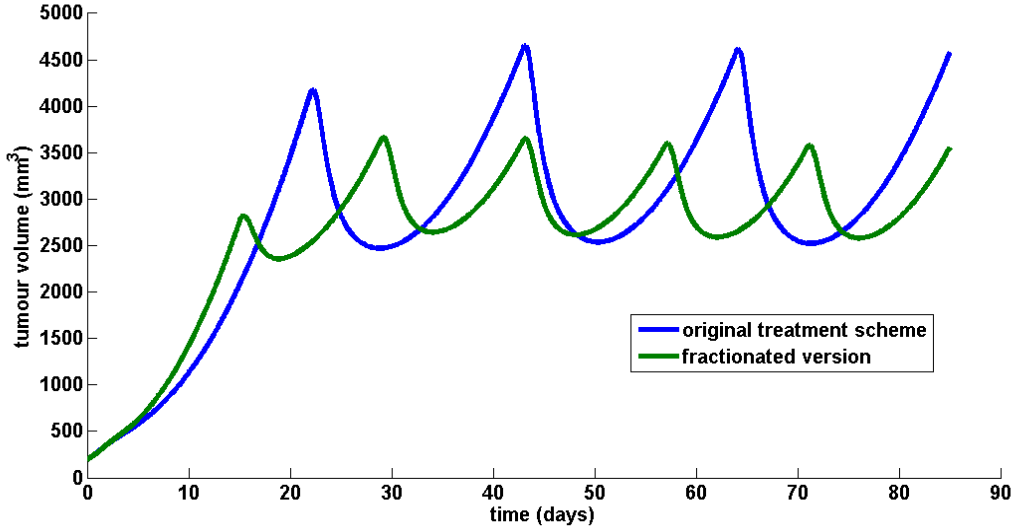


Figure 4-3: Breast cancer Oncosimulator: tumor volume time evolution for the two simulated bevacizumab schemes.

Input parameter	**Sim1**	**Sim2**	**Sim3**
Actinomycine administration time point 1 (days)	1	1	1
Actinomycine administration time point 2 (days)	15	15	17
Vincristine administration time point 1 (days)	1	1	1
Vincristine administration time point 2 (days)	8	10	10
Vincristine administration time point 3 (days)	15	15	17
Vincristine administration time point 4 (days)	22	22	24

Table 4-1: Wilms tumor Oncosimulator: input data for the different delay cases.

Result	Sim1	Sim2	Sim3
Percentage of tumor volume change	16.73	14.66	2.3
Percentage of proliferating cells out of the total tumor cells in the initial tumor	15.8		
Percentage of dormant cells out of the total tumor cells in the initial tumor	19.49		
Percentage of differentiated cells out of the total tumor cells in the initial tumor	59.02		
Percentage of dead cells out of the total tumor cells in the initial tumor	5.69		
Growth fraction of the initial tumor	16.75		
Percentage of proliferating cells out of the total tumor cells in the final tumor	9.63	9.41	7.94
Percentage of dormant cells out of the total tumor cells in the final tumor	14.54	14.42	18.11
Percentage of differentiated cells out of the total tumor cells in the final tumor	69.12	69.46	66.85
Percentage of dead cells out of the total tumor cells in the final tumor	6.71	6.71	7.1
Growth fraction of the final tumor	10.33	10.09	8.55

Table 4-2: Wilms tumor Oncosimulator: result data for the different delay cases.

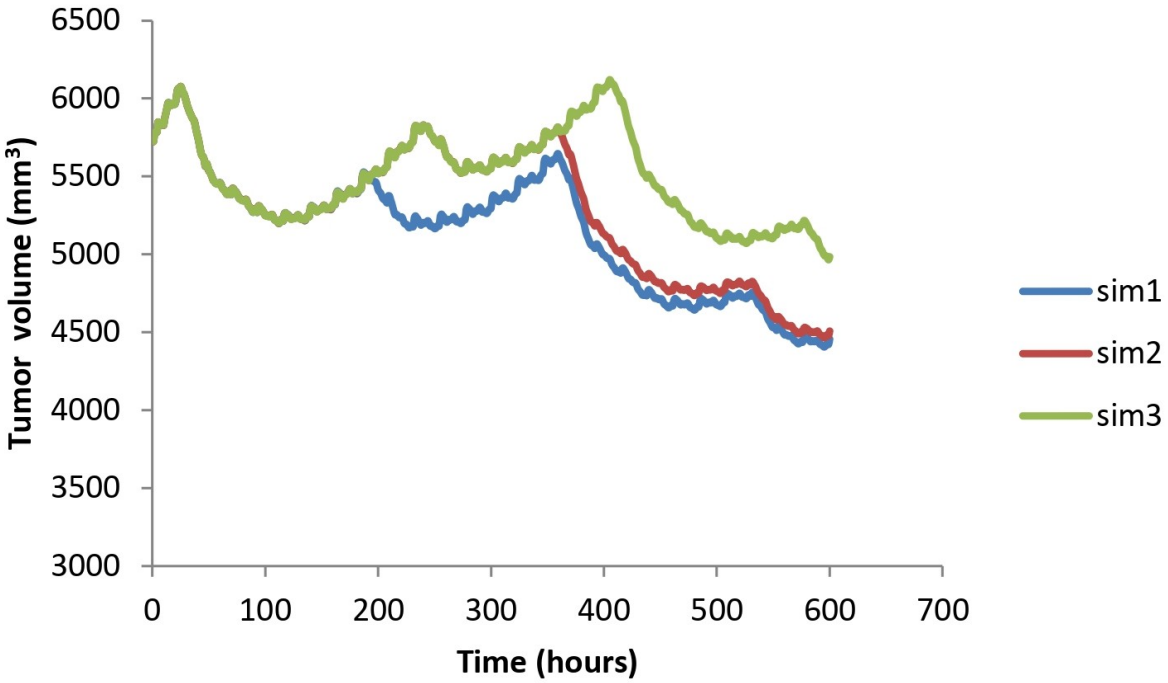


Figure 4-4: Wilms tumor Oncosimulator: tumor evolution graph over time for all three simulated delay cases.

4.4 References

1. Idelhauser J, Humm BG, Beez U. A Clinical Decision Support System for Personalized Medicine. In: *Proceedings of the Collaborative European Research Conference (CERC 2016)*. Cork, Ireland: CERC; 2016.

2. Stamatakos G, Dionysiou D, Lunzer A, et al. The technologically integrated oncosimulator: combining multiscale cancer modeling with information technology in the in silico oncology context. *IEEE J Biomed Health Inform*. 2014;18(3):840–54. doi: 10.1109/jbhi.2013.2284276.

3. Clinical Decision Support Systems. PSNet [internet]. Rockville (MD): Agency for Healthcare Research and Quality, US Department of Health and Human Services. 2019.

4. Bucur, A., van Leeuwen, J., Christodoulou, N. *et al.* Workflow-driven clinical decision support for personalized oncology. *BMC Med Inform Decis Mak* 16 (Suppl 2), 87 (2016). https://doi.org/10.1186/s12911-016-0314-3

5. Java Native Interface Specification https://docs.oracle.com/en/java/javase/17/docs/specs/jni/index.html

6. Sutton RT, Pincock D, Baumgart DC, Sadowski DC, Fedorak RN, Kroeker KI. An overview of clinical decision support systems: benefits, risks, and strategies for success. NPJ Digit Med. 2020 Feb 6;3:17. doi: 10.1038/s41746-020-0221-y. PMID: 32047862; PMCID: PMC7005290.

7. Williamson K. The Role of Clinical Decision Support Systems in Informed Medical Decision-Making. American Hospital & Healthcare Management. Accessed July 1, 2025. https://www.americanhhm.com/articles/the-role-of-clinical-decision-support-systems-in-informed-medical-decision-making

8. Stamatakos G, Kolokotroni E, Panagiotidou F, Tsampa S, Kyroudis C, Spohn S, Grosu A-L, Baltas D, Zamboglou C and Sachpazidis I (2025) *In silico* oncology: a mechanistic multiscale model of clinical prostate cancer response to external radiation therapy as the core of a digital (virtual) twin. Sensitivity analysis and a clinical adaptation approach. *Front. Physiol.* 16:1434739. doi: 10.3389/fphys.2025.1434739

9. Meyerheim M, Panagiotidou F, Georgiadi E, Soudris D, Stamatakos G and Graf N (2025) Exploring the *in silico* adaptation of the Nephroblastoma Oncosimulator to

MRI scans, treatment data, and histological profiles of patients from different risk groups. *Front. Physiol.* 16:1465631. doi: 10.3389/fphys.2025.1465631

10. Poleszczuk, J., M. Bodnar, U. Foryś. 2011. New approach to modelling of antiangiogenic treatment on the basis of Hahnfeldt et al. model. Math Biosci Eng 8(2): 591-603. doi:10.3934/mbe.2011.8.591

11. Argyri, K., D. Dionysiou, and G. Stamatakos (2012) Modelling the interplay between pathological angiogenesis and solid tumor growth: The anti-angiogenic treatment effect. Advanced Research Workshop on In Silico Oncology and Cancer Investigation - The TUMOR Project Workshop (IARWISOCI), 2012 5th International, 22-23 Oct. 2012, vol., no., 1,4

12. Kalinin K. Implementing Clinical Decision Support Systems: A Practical Guide. Topflight Apps. Published May 16, 2025. Accessed July 1, 2025. https://topflightapps.com/ideas/clinical-decision-support-system-implementation/

13. Kao D, Larson C, Fletcher D, Stegner K. Clinical Decision Support May Link Multiple Domains to Improve Patient Care: Viewpoint. JMIR Med Inform. 2020 Oct 16;8(10):e20265. doi: 10.2196/20265. PMID: 33064106; PMCID: PMC7600021.

14. Mosavi NS, Santos MF. Enhancing Clinical Decision Support for Precision Medicine: A Data-Driven Approach. *Informatics*. 2024; 11(3):68. https://doi.org/10.3390/informatics11030068

5. Standardization of Clinical Questions

A clinical question can be defined as a structured inquiry designed to guide medical practice and research, often focusing on a patient's condition, intervention, comparison, and outcomes. Clinical questions' involvement in personalized medicine are crucial because they allow health professionals to take into account an individual's biological and lifestyle factors that would be the basis for a unique diagnosis, intervention, or preventive measure suggestion. The complexity and specificity required in personalized medicine, leading to incorporate individual genetic make-up, lifestyle, and environmental exposures to result in targeted interventions, clearly indicate the critical importance of standardizing said clinical questions [1]. Standardization would ensure consistency of data collection, improve data interoperability across systems, enhance the accuracy of clinical decision support, and ultimately accelerate robust research, allowing for the reliable comparison and synthesis of evidence to advance individualized care [2].

In this chapter the different aspects of clinical question standardization are briefly explained and a proposed infrastructure is described, which can operate within the confines of a generic web application hosting and handling simulation models and their pertinent data, thereby acting as an extension of the basic infrastructure as described in chapter 2.

5.1. The overall need for standardization

Data standardization significantly enhances the functionalities of biomedical information systems, especially CDSS, which, utilizing standardized data input, can more efficiently and accurately process and analyze patient information. The provided consistency facilitates the creation of robust evidence-based rules and algorithms which in turn results to more effective interpretation of individual patient data. Therefore a CDSS can provide features at the point of care such as more timely and pertinent alerts regarding potential drug interactions or allergies, more accurate diagnostic suggestions, and recommend treatment options that are specifically tailored to individual needs [2].

Beyond CDSS, other types of biomedical information systems can benefit from data standardization. A prime example is electronic health records (EHRs); data standardization allows them to become more efficient in data retrieval, information exchange between different healthcare providers and analysis of medical findings (tests, model simulations, etc.) for research purposes [3]. Data standardization plays a crucial role for the integration between diverse biomedical databases and knowledge resources, overcoming differences in schema, development stack, or general representation. This enhances personalized healthcare by offering the stakeholders an environment where infrastructure and information are more interconnected, and thus more available and comprehensive. To this end, the use of standardized questions can improve the performance of biomedical question answer systems or modules, since they can allow them to query for and retrieve more pinpointed, well structured and properly bundled data that can allow for faster and more personalized medical decisions. The “common language” notion imported by standardization, can also help for a more holistic understanding of the differently structured patient profiles, and the high level inherent information that can be extracted by their study (behavioral, social, etc.)

With respect to clinical research and knowledge discovery, clinical question standardization enables researchers to conduct more robust meta-analyses, acting as a conceptual integration layer (either adding to an existing integration in data layer, or replacing it where necessary) for data inquiry and collection [4]. Standardization also supports the development and utilization of common data models (CDMs), which provide a unified structure for representing patient data across multiple institutions and research networks, fostering collaborative research efforts [5]. This uniformity in data collection and representation ultimately improves the efficiency and reliability of clinical trials designed to evaluate the effectiveness of personalized therapies [6]. The overall result is the acceleration of research pace in personalized medicine, thereby enabling the identification of novel biomarkers, the discovery of new therapeutic targets, and the development of more effective and tailored treatment strategies for individual patients [4].

A lack of proper standardization can make it challenging for a CDSS to accurately interpret patient information and provide reliable advice due to inconsistent data. This can lead to "alert fatigue" and potentially inaccurate or less effective clinical recommendations. [7]. Research based on non-standardized questions can be prone to limitations in generalizability and reproducibility, due to the overhead added to properly translate data between their source-related representations. This hinders the validation procedure of research findings through replication and consequently the translation of those findings into widespread clinical practice [8]. This furthermore impacts personalized medicine, since the essential large data-sets required to train and validate sophisticated artificial intelligence and machine learning models cannot be easily created [9]. In addition, the creation of data silos will isolate valuable patient information and prevent the identification of crucial patterns.

5.2. Existing tools and technologies for proper standardization

On a conceptual level clinical question and answer standardization can be achieved in four different levels, using specialized tools for each level. For clinical terms themselves, structured vocabularies such as SNOMED CT (Systematized Nomenclature of Medicine – Clinical Terms) [10], LOINC (Logical Observation Identifiers Names and Codes) [11], and RxNorm [12] can provide consistent and unambiguous representations. This leads to questions having their terms mapped in a uniform manner which ensures that the meaning of clinical concepts is consistent across different systems and among various users. For the relationship and hierarchies between different clinical terms, biomedical ontologies can be used. This can group questions based on the levels of abstraction assigned to their medical terms. Therefore relationships between questions can be formulated, the answers of which can be valuable in multiple cases (e.g. comorbidity or drug interaction questions.

To further place clinical questions into relevant workflows for clinical practice, frameworks such as PICO (Patient, Intervention, Comparison, Outcome) [13] and PICOT (Patient, Intervention, Comparison, Outcome, Timeframe) [14] can be used. By consistently applying these frameworks, key elements of a personalized clinical scenario are captured in a structured manner, including a clear definition of the patient or problem, the intervention being considered, the comparison intervention (if any), the desired outcome, and the

relevant timeframe and setting. This can add another abstraction layer that can allow the handling of a clinical case and/or make it research relevant in cross-platform environments. Finally, since proper presentation to the clinician is of paramount importance, Common Data Models (CDMs), such as the Observational Medical Outcomes Partnership (OMOP) [15] and the Clinical Data Interchange Standards Consortium (CDISC) [16] standards can be used. These data models provide a consistent structure for capturing the complex information associated with clinical questions and their answers in personalized medicine, including relevant patient characteristics (such as genetic information), specific treatment details, and the observed outcomes. [5].

Also crucial for day-to-day operations is the proper and efficient storage of this information from questions to workflows and presentation data structures to reduce access time, computational resources and enhance reusability, which can benefit more general systems such as CDSS, which rely on these entities. Designing relational databases that are specifically tailored to accommodate structured clinical questions and answers is a crucial first step [17]. This involves defining clear entities (such as patients, questions, interventions, outcomes, etc.), specifying their relevant attributes, and establishing the relationships between them [18]. Additional mechanisms for indexing and cataloging these resources in a way that allows for easy search and retrieval based on various criteria, such as the PICO components or relevant keywords can also be defined using known tools such Elasticsearch [19] or Apache Solr [20]. Finally, auditing tools applicable to the persistence layer can provide a version control of the stored data, to ensure their integrity and readability.

So far, numerous organizations and initiatives are actively engaged in the development and promotion of healthcare data standards, which indirectly contribute to the standardization of clinical questions and answers in personalized medicine. Key organizations such as HL7 International, and the aforementioned CDISC, LOINC, and SNOMED are at the forefront of developing standards for data exchange, clinical trial data, laboratory observations, and clinical terminology, respectively [21]. Initiatives like the NIH Common Fund Data Ecosystem (CFDE) aim to modernize data sharing, integration, and standardization across the biomedical research community [22]. The Health Data Standardization Taskforce has been established to specifically address interoperability through standardization in various healthcare settings [23]. While these efforts may not be exclusively focused on standardizing clinical questions, the development of standardized terminologies, data models, and exchange formats provides the essential building blocks for achieving standardization in clinical inquiry and response within personalized medicine.

5.3. Implementation Challenges

Implementing or updating a medical system based on clinical question and answers can present a set of challenges either from a technical perspective or by its potential effect of the day-to-day operations of the institution that will utilize such a system. A lack of proper documentation and training material can lead to resistance, which is part of the natural course for health professionals whenever standardized processes and technologies

are introduced in an institution [24]. Factors such as heavy workloads make perceived time scarce, inadequate training on the new developments, irrelevance to the day-to-day exercise, and possible disruption of the established clinical workflows maybe causes of this resistance [25]. Proper update of all involved personnel on the new features and results of a standardization procedure which will include a clear demonstration of the tangible benefits that are brought for both healthcare professionals and their patients is required to change this.

Data entry can also present problems, not only in the case of inadequate training which can lead to human errors but also if there is some missing initial input for some clinical cases, of the gathered information is incompatible due to the heterogeneity between different healthcare providers and institutions, for example in cases of patients that for some reason have chosen to see different doctors from case to case. Extensive data validation and sanitization is required to not only keep the stored information relevant between its own entities, but to avoid cases of "dirty" (incomplete or otherwise corrupted) data, as well as for security concerns. Legislation can also help by determining the forms and structures of the exchanged data and implementing comprehensive ongoing quality assurance processes [7].

Moreover, the usage of more sophisticated simulation models, clinical processes and detailed medical data, ranging from molecular to organ and system biocomplexity levels, unavoidable leads to the ever-increasing amount of sensitive personal data, such as genetic information. This requires not only legislative support to ensure that the patient is properly informed and has duly consented to the utilization of their own data, but powerful infrastructures to secure such systems from data leakage, thereby ensuring the necessary privacy. VPNs, cloud infrastructures and carefully developed applications with as less software vulnerabilities as possible is of crucial importance to the point of requiring specialized trained professionals outside the fields of clinical researchers or software developers. Included in the security requirements are of course the proper data anonymization and encryption within the persistence layer, usually in compliance with legal frameworks such as GDPR [26].

5.4. Conceptual system overview

The proposed module for handling clinical questions and their answers is to currently serve as an extension to the infrastructure presented in chapter 2. It is a part of the data repository which resides in the persistent layer, as depicted in figure 2-2. It is coupled to the basic model entities and its aim is to classify execution results and translate them into valuable clinical insight, on which personalized medical decisions can be made. Its main data input are model output parameter values, and although the overall design allows it to accept any type of numerical input and map it to a set of predetermined string values based on value ranges, the additional context its current version requires, elevates its role to more than a simple numerical value "translator". The overall goal is to define sets of clinical questions and lists of corresponding answers, usable by the model execution engine so that additional information and value to model execution result reports can be added.

Figure 5-1 presents the functionality of the module with respect to the other parts of the infrastructure. It operates in close collaboration with the model repository to map clinical answers to certain model output variables, and it uses these mappings to store not only the clinical answers, but to properly formulate the model execution data with the related clinical questions and their corresponding answers and present them when required. The system operates under a specific set of assumptions about the definition and usage of the stored clinical questions:

- A clinical question has multiple clinical answers. Each answer is defined separately. Answers have their own table with an n-to-1 relationship with respect to the question table.
- A clinical answer is connected to a model via only one of its output parameters. It is mapped to either a single parameter value, or a certain range of values. This creates two distinct n-2-n relationships, since the clinical answer table acts as a connection table between the clinical questions and the models, and also the clinical questions and the parameters.
- Model executions include information about input and output parameters. For each output parameter the value, the related clinical questions and the proper answers based on the value are also stored. Each combination of parameter-question-answer is stored in a separate row.

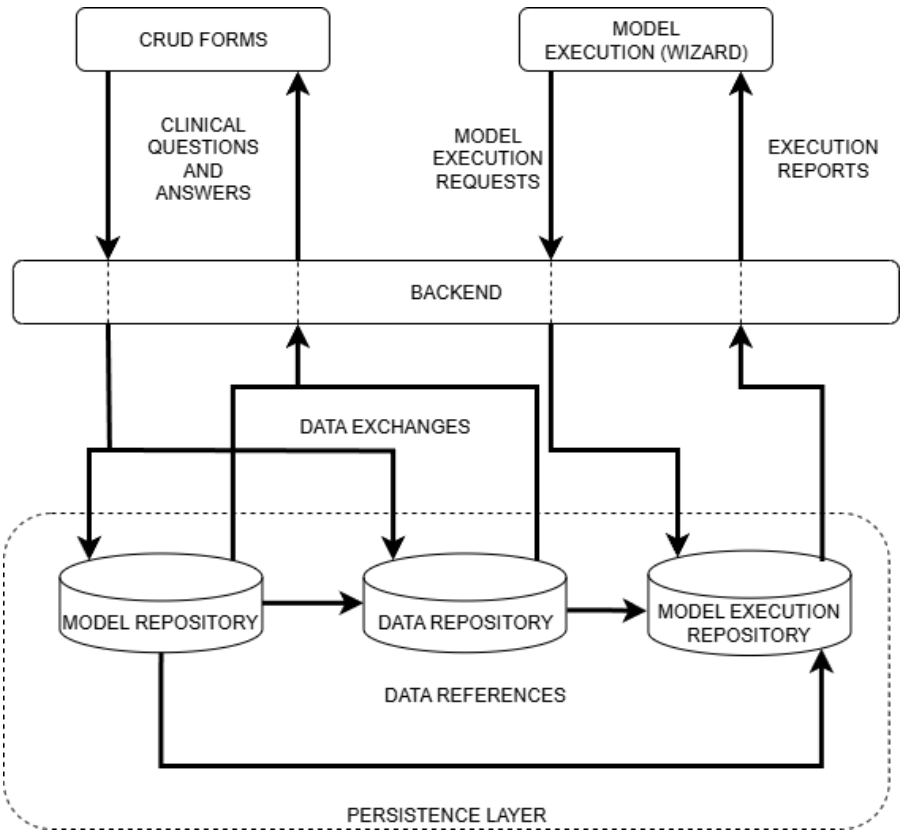


Figure 5-1: Functionality diagram of the proposed clinical question module.

5.5 User workflow

The user workflow is shown in Figure 5-2. It is applicable to all users (modelers and clinicians) and it refers to two specific functionalities, the CRUD handling of clinical questions and answers, plus the utilization of the stored entities by the system when a model execution report is requested.

After logging to the system, the user should reach the proper CRUD forms. To upload a clinical question and its answers for storage, data must be entered in at two tables. First the clinical question table, where the actual question and a secondary description (if necessary) can be given, followed by the uploading of all possible answers to that question.

To properly upload an answer, the user must choose a model to for which the clinical question will apply to. Then, they must define an output parameter of the chosen model and set a fixed value or a value range. After that, they must provide a text representation of the clinical answer. In the same way more than one answers can be defined for the same question based on the different parameter values. The goal is that, for an execution of a chosen model, any stored clinical questions will be answered accordingly based on the output value of the corresponding parameters. The results will be stored in the database and retrieved accordingly.

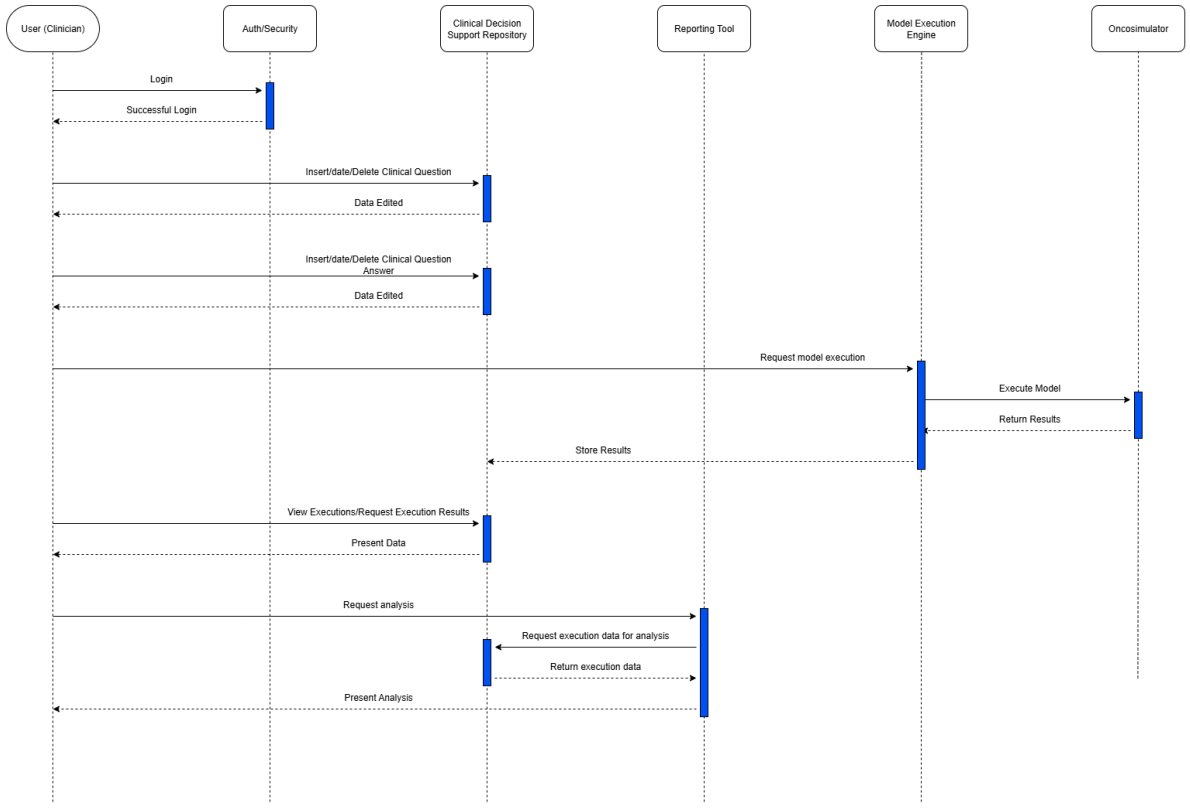


Figure 5-2: User workflow

Based on the aforementioned functionality, when a model execution is requested, then the system calls to the OS to execute the model. After a successful execution there are several output parameter values to be stored in the database. At that point in time, the back

end is responsible for retrieving any clinical questions and answers related to the model. After checking the output values and determining the proper answers to be used, it stores in the model execution repository the combined information about the model, its execution, the outputs, the clinical question ids and the clinical answer ids. When a user requests the report of a model execution, the back end retrieves the stored data, and with them, the textual information of all the clinical questions and answers. A proper report is created and presented in the GUI, with the option to be downloaded.

5.6 References

1. Stillwell, Susan B. DNP, RN, CNE; Fineout-Overholt, Ellen PhD, RN, FNAP, FAAN; Melnyk, Bernadette Mazurek PhD, RN, CPNP/PMHNP, FNAP, FAAN; Williamson, Kathleen M. PhD, RN. Evidence-Based Practice, Step by Step: Asking the Clinical Question: A Key Step in Evidence-Based Practice. AJN, American Journal of Nursing 110(3):p 58-61, March 2010. | DOI: 10.1097/01.NAJ.0000368959.11129.79

2. Musen, M.A., Middleton, B., Greenes, R.A. (2021). Clinical Decision-Support Systems. In: Shortliffe, E.H., Cimino, J.J. (eds) Biomedical Informatics. Springer, Cham. https://doi.org/10.1007/978-3-030-58721-5_24

3. Institute of Medicine (US) Committee on Data Standards for Patient Safety; Aspden P, Corrigan JM, Wolcott J, et al., editors. Patient Safety: Achieving a New Standard for Care. Washington (DC): National Academies Press (US); 2004. 4, Health Care Data Standards. Available from: https://www.ncbi.nlm.nih.gov/books/NBK216088/

4. Pétavy F., Seigneuret N., Hudson L.D., Becnel J.B., Becnel L.B., Galvez J., Jackson T., Kush R.D., Harmon N., Zhou F.L., et al. The Turning Point for Clinical Research: Global Data Standardization, accessed on 21 April 2025, https://www.appliedclinicaltrialsonline.com/view/turning-point-clinical-research-global-data-standardization .

5. Feldman, K., Johnson, R.A. & Chawla, N.V. The State of Data in Healthcare: Path Towards Standardization. *J Healthc Inform Res* 2, 248–271 (2018). https://doi.org/10.1007/s41666-018-0019-8

6. Collins FS, Varmus H. A new initiative on precision medicine. N Engl J Med. 2015 Feb 26;372(9):793-5. doi: 10.1056/NEJMp1500523. Epub 2015 Jan 30. PMID: 25635347; PMCID: PMC5101938.

7. Muhiyaddin R, Abd-Alrazaq AA, Househ M, Alam T, Shah Z. The Impact of Clinical Decision Support Systems (CDSS) on Physicians: A Scoping Review. Stud

Health Technol Inform. 2020 Jun 26;272:470-473. doi: 10.3233/SHTI200597. PMID: 32604704.

8. Cruz-Correia, R., Ferreira, D., Bacelar, G. *et al.* Personalised medicine challenges: quality of data. *Int J Data Sci Anal* 6, 251–259 (2018). https://doi.org/10.1007/s41060-018-0127-9

9. Goetz LH, Schork NJ. Personalized medicine: motivation, challenges, and progress. Fertil Steril. 2018 Jun;109(6):952-963. doi: 10.1016/j.fertnstert.2018.05.006. PMID: 29935653; PMCID: PMC6366451.

10. SNOMED CT (Systematized Nomenclature of Medicine – Clinical Terms) Official Website: https://www.snomed.org/

11. LOINC (Logical Observation Identifiers Names and Codes) Official Website: https://loinc.org/

12. RxNorm National Library of Medicine (NLM) RxNorm homepage: https://www.nlm.nih.gov/research/umls/rxnorm/

13. Eldawlatly A, Alshehri H, Alqahtani A, et al. Appearance of Population, Intervention, Comparison, and Outcome as research question in the title of articles of three different anesthesia journals: A pilot study. *Saudi J Anaesth*. 2018;12(2):299-301. doi:10.4103/sja.SJA_767_17. PMCID: PMC587521 https://pmc.ncbi.nlm.nih.gov/articles/PMC5875219/

14. University Libraries/Evidence-Based Practice Guides: The Claude Moore Health Sciences Library link for PICO also includes PICOT: https://guides.hsl.virginia.edu/c.php?g=921177&p=6638623

15. OMOP (Observational Medical Outcomes Partnership) Official Website (OHDSI - Observational Health Data Sciences and Informatics): https://www.ohdsi.org/

16. CDISC (Clinical Data Interchange Standards Consortium) Official Website: https://www.cdisc.org/

17. Park Y, Yoon Y, Koo H, Yoo S, Choi C, Beck S, Kim T. Utilization of a Clinical Trial Management System for the Whole Clinical Trial Process as an Integrated Database: System Development. J Med Internet Res 2018;20(4):e103. URL: https://www.jmir.org/2018/4/e103. DOI: 10.2196/jmir.9312

18. Niazi U, Stuart C, Soares P, Foure V, Griffiths G. An open-source SQL database schema for integrated clinical and translational data management in clinical trials. Clin Trials. 2025 Jun;22(3):374-377. doi: 10.1177/17407745241304331. Epub 2024 Dec 25. PMID: 39720844; PMCID: PMC12092935.

19. Elasticsearch Official Website (Elastic): https://www.elastic.co/products/elasticsearch

20. Apache Solr Official Website (Apache Solr): https://solr.apache.org/

21. Pétavy F, Seigneuret N, Hudson LD, et al. Global Standardization of Clinical Research Data. *Appl Clin Trials*. 2019;28(4):18-24.

22. NIH Common Fund Data Ecosystem (CFDE) Official Website: https://commonfund.nih.gov/dataecosystem

23. Kwon A, Lee HY, Shin SY, Yang K, Sung Y, Lee K, Byeon NS, Lim TH, Lee JH. Current Health Data Standardization Project and Future Directions to Ensure Interoperability in Korea. Healthc Inform Res. 2024 Apr;30(2):93-102. doi: 10.4258/hir.2024.30.2.93. Epub 2024 Apr 30. PMID: 38755100; PMCID: PMC11098766.

24. Blobel, B. (2017). Standardization for Mastering Healthcare Transformation–Challenges and Solutions. *European Journal for Biomedical Informatics*, *13*(1), 9-15.

25. Lv F. From Evidence to Action: Addressing Challenges in Implementing EBP in Nursing. J Evid Based Med- Healthc 2024;11(02):1-2.

26. GDPR (General Data Protection Regulation) Official Text/Information Portal: While the official legal text is published in the Official Journal of the European Union, a widely used and accessible resource is: https://gdpr-info.eu/

6. The Hyperion Application

Towards implementing the fundamental principles for developing a computational infrastructure to host simulation models and their associated data, as explained in Chapters 2 and 5, and in accordance with the objectives of this work, the Hyperion application has been developed. It's a web application primarily designed to enable stakeholders to store simulation models by way of uploading their specific characteristics. It also empowers both modelers and clinicians to execute these models and to view and manage the results of these executions through the application's various modules.

This endeavour aims to consolidate core research material and manage the knowledge generated from its utilization. The goal is to benefit both modelers by streamlining their day-to-day processes for the adaptation and validation of their simulation models, and clinical physicians by reducing response times for their patients. Simultaneously, it allows for the provision of case-specific medical advice, thereby contributing to the faster and more successful implementation of the concept of personalized medicine. Patients, as the ultimate recipients of medical services, will see an improved quality of life. This improvement stems from a reduction in the time and effort required to establish and implement the appropriate medical procedure for their case, and from the effort's contribution to minimizing the side effects of their personalized treatment by predicting and evaluating various potential outcomes from a range of therapeutic regimens.

The following subsections provide a general overview of the application and the requirements needed for its development. They will also successively present the main sub-units, along with their significance and contribution to the overall system.

6.1. Overview

Hyperion is designed with a modular approach. Each part of a simulation model's life cycle from creation to clinical practice is handled in a certain application module. Although the modules are not required to be strongly coupled, they do share a connection with the main module, which is the model repository. This module holds the main model data, upon which the other modules can operate. These data are also the starting point for any perceived expansions, such as the examples of chapters 3 and 4. The main conceptual goal of the application is to give a clear and concise, albeit scalable and extendable answer to what is perceived by this work as the main go-to query:

"If we were to view simulation models solely as black boxes that simply accept and return data, then what is the maximum that we can do with them?"

Figure 6.1 gives the architectural diagram of Hyperion. Each module can be perceived to be its own full stack entity, although the implementation details can differ, while the user experience through proper front-end programming can remain unified and implementation agnostic.

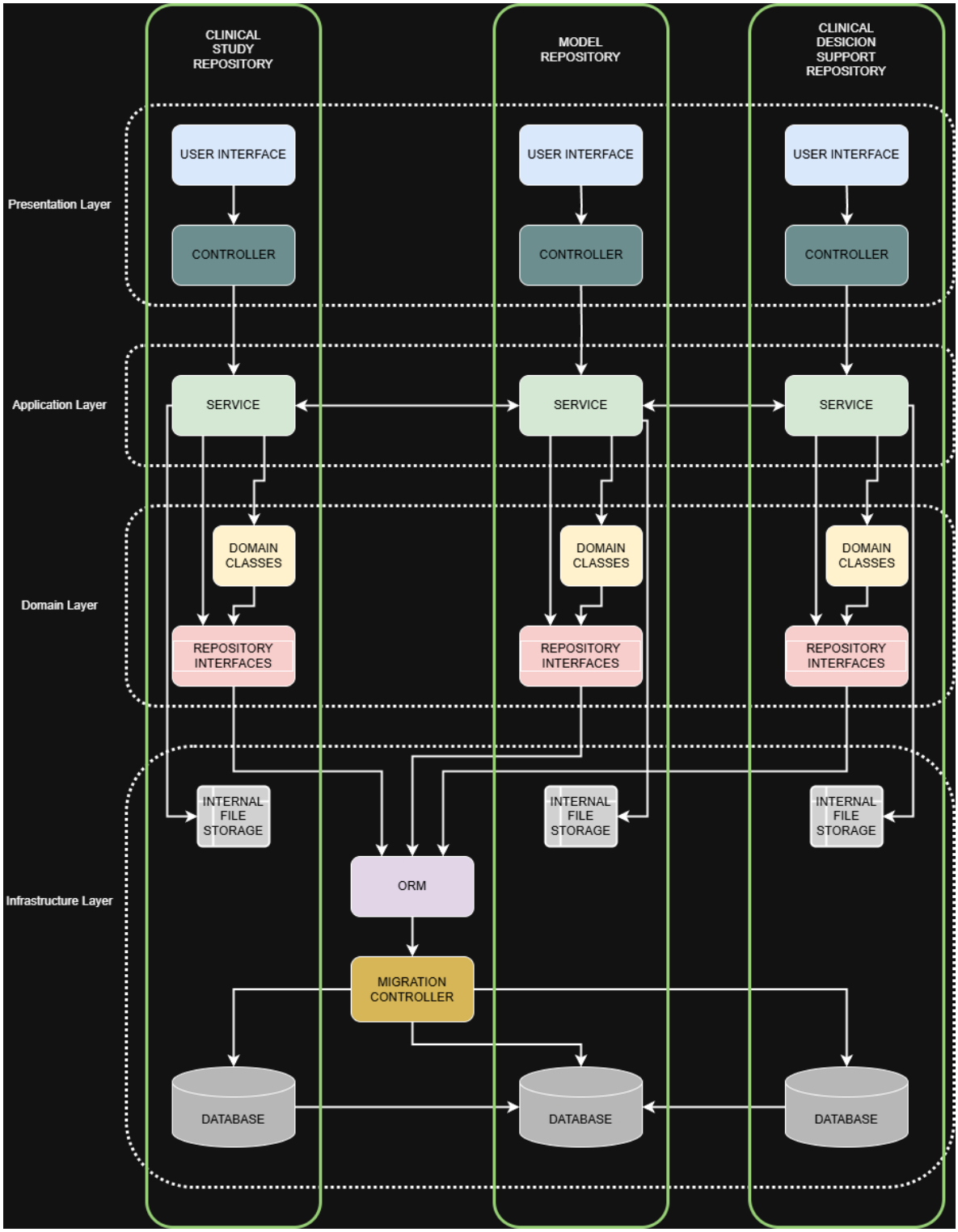


Figure 6-1: Hyperion layered architectural diagram

6.1.1. Implementation Details

The implementation stack of Hyperion is comprised of MySQL, MyBatis, Flyway, Spring Boot, Java, and React. For the separation and management of the code for the different entities, a Domain-Driven Design (DDD) [1] approach was chosen to allow

flexibility and scalability as new modules can enter the system. The choice of the design was made keeping in mind not only the current monolithic implementation but also aiming to be suitable for migration to a microservice architecture.

For the persistence layer, the free and open-source version of MySQL [2] is chosen. It is a popular reliable relational database management system (RDBMS) offering ACID-compliant transactions, strong data consistency, and high performance for relational data workloads. For object-relational mapping, the system utilizes MyBatis [3]. Unlike more automated ORM tools such as Hibernate [4], MyBatis allows for greater control of the utilized SQL queries by allowing to write their full implementation. The use of dedicated XML mappers per entity repository further complies with Domain-Driven design principles. In order to keep track of database updates and migrations, Flyway [5] is used. With a system based on sequentially versioned SQL scripts it provides a reliable system history and overall schema consistency.

The back end application layer is built using Spring Boot [6] and Java [7]. Spring Boot accelerates development by simplifying project configuration and bean control, shifting away from the traditional use of XML files used by Spring. It still supports features such as dependency injection, transaction management, and RESTful API development. Java, is a well known, all-purpose, programming language that offers reliability, scalability, and cross-platform portability.

The presentation layer uses React [8], a widely used JavaScript library for building user interfaces. React's component-based architecture promotes reusability, modularity, and ease of maintenance. Its unidirectional data flow and virtual DOM implementation enable efficient rendering and responsiveness for rich client-side experiences.

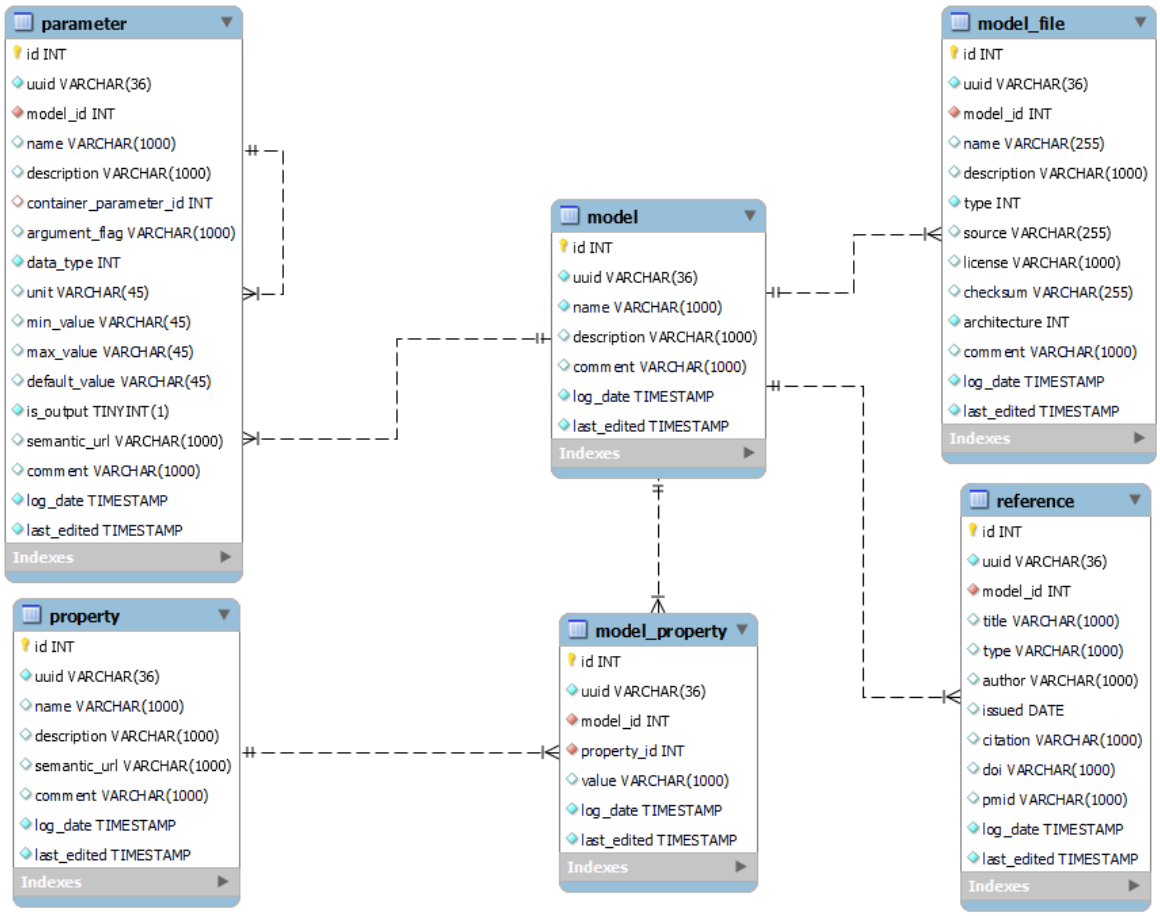
The code structure that handles any entity within a repository is based on the diagram of figure 6-1. A set of pages/screens in the user interface will call a set of APIs which reside in the back end controllers. The controllers will relay the data to the services, which in turn contain all the logic that is needed to use the sent data and return the requested results, such as data transformation functions and/or communication with other services to exchange data. This is necessary, as each service (ideally) calls only its own corresponding repository interface which in turn is responsible for querying the underlying database.

6.2. Model Repository

The model repository is Hyperion’s backbone. It is the module that handles the basic entity around which the entire system is build, which is the simulation model. As shown in figure 6-2, the repository is comprised of the following tables

- **Model**: This table stores the basic descriptive information about the model, and is the point of reference for the entire application, since it is the table with the most foreign key references throughout Hyperion. Both clinical study and clinical decision support repository are based on the model id to organize their data.
- **Parameter**: The repository’s second most important table, it holds the full descriptive and functional information about a model’s parameters. It is used from all the other modules, as it is the basis not only for formulating proper execution

commands of a model, but also acts as a blueprint for defining clinical questions, clinical study subjects and formulating execution reports. This table also holds a reference to itself; in several cases a simulation model either receives or produces its values regarding one or more of its parameters in files. Therefore, a file type parameter can be defined, and through the table's foreign key to itself the rest of the contained parameters can refer to the file as its "container" parameter. This further facilitates report formulation, since the back end, being aware of the inter-parameter relationships, and assuming the existence of some well-defined conventions about the file structure, can automatically "unpack" the required values from their "containers" and store them into the proper tables of the clinical decision support repository.



- **Figure 6-2:** Model repository ER diagram

- **File**: This table holds the information about a model's files. Files of various types (executable, text, image, supplementary scripts, etc.) can be uploaded via the user interface. The files stored in the application's internal file storage system under the path {rootFolderPath}/models/{modelUuid}
- **Property**: Simulation models can be classified depending on their computational methods, biocomplexity level and a number of other perspectives [9]. This table

holds the classification descriptive information, while the model-property table holds the information regarding the many-to-many connection with the model table.

6.3. Clinical Study Repository

Although the Model Execution Wizard provides the capability for individual model executions in order to assess single patient cases, there is also a need to record and collectively evaluate multiple executions of a model. This requires grouping and managing results for further clinical and statistical processing. For this purpose, the Clinical Study Repository has been created, the ER diagram of which is shown in Figure 6-3. The main principles governing this schema are as follows:

- Each clinical study can involve one and only one simulation model.
- A clinical study consists of a set of anonymized patients.
- A stratification is defined based on a model and one of its parameters. Since the definition uses a specific value or a range of values for the parameter, it is possible for a parameter to participate in multiple stratifications, something that is expected in the context of a clinical study.
- An anonymized patient may belong to one or more stratifications.
- An anonymized patient has at least one and maybe more instances. Each instance has its own set of input and output parameters. This instance is called a subject.
- Each model execution involves up to one specific subject.
- If a model execution does involve an anonymized patient but no specific subject, a blank subject is automatically created.

Given the above principles, the workflow for recording a clinical study can also be deducted. The user can select the model, define the desired stratifications based on the parameters they want, and then enter into the system a set of anonymized patients. They can then specify the stratifications each patient will belong to and the different instances (subjects) for each patient. After executing the model (since executions are recorded along with the subject they concern), it is possible to retrieve the relevant data to proceed with any analysis. A detailed example will be provided in the next chapter.

The tables of the Clinical Study Repository are as follows:

- **Clinical Study:** This is the main table of the repository, containing the descriptive information for each clinical study.
- **Anonymized Patient:** This table includes a one-to-one correspondence with all real patients in a clinical study, storing information that does not conflict with personal data protection regulations. Specifically, it includes only a Patient ID field, which according to the predetermined rules of each clinical study may be either anonymized or pseudonymized. Additionally, it records data such as gender, height, weight, and initial tumor stage, which are necessary for evaluating the patient's condition before performing any model executions.
- **Stratification:** This table records all possible patient stratifications. It includes the descriptive information, the related model and parameter IDs, and the parameter

value definitions (either as a default value or a value range). Since each patient can belong to more than one category, a many-to-many relationship is created between the Stratification and Anonymized Patient tables. For this reason, there is an intermediate table called Anonymized Patient Stratification.

- **Subject:** This table holds all information about a patient instance, including the related stratification and model IDs. Due to the entity's fundamental definition, a double many-to-many relationship is also established with the Model Repository's Parameter table, using the next two tables.
- **Subject Parameter:** Since each parameter can be involved in different subjects, and each subject is related to many parameters, this table also includes an additional value field, which enables the reconstruction of a subject after model execution by retrieving the values of the input and output parameters. This facilitates the formation of the report of a specific execution, also addressing any clinical questions related to the model, and can be used for further statistical analysis, as will be discussed in a later subsection.

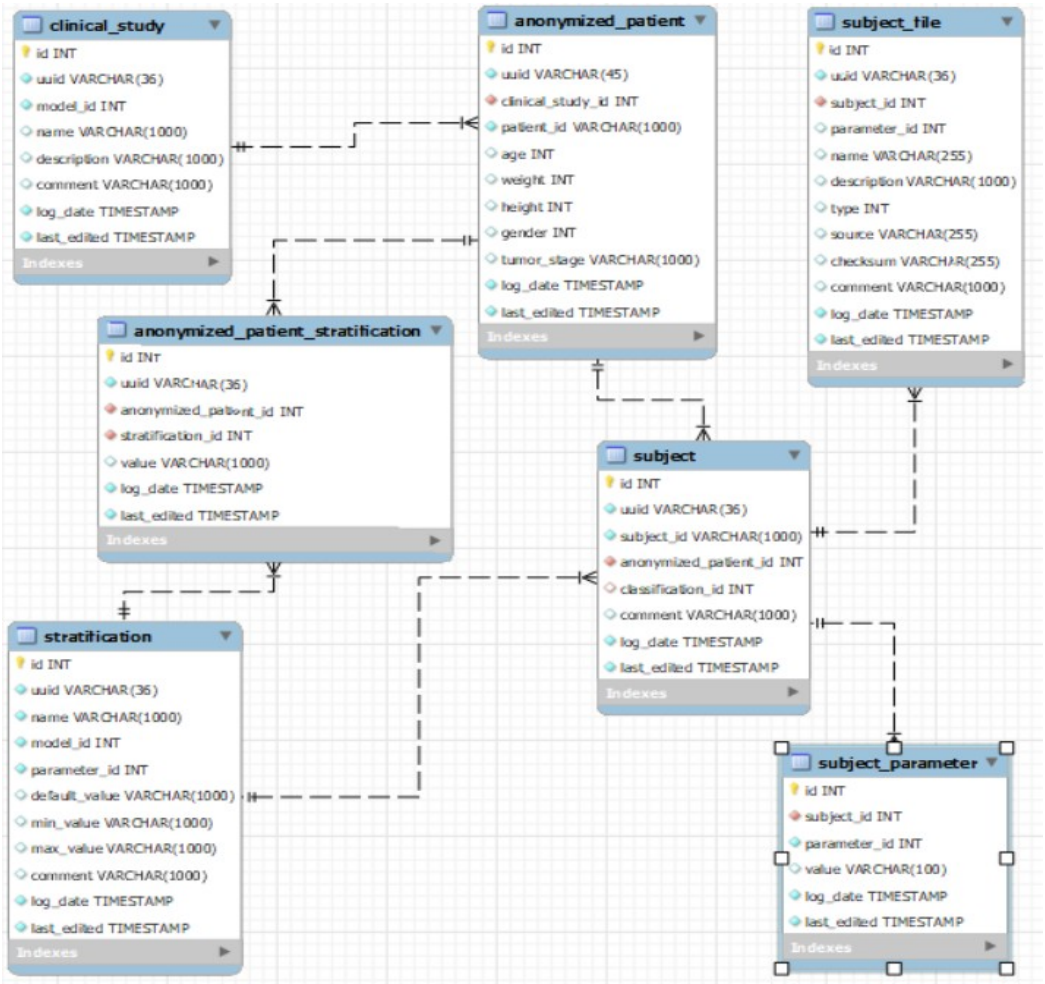


Figure 6-3: Clinical study repository ER diagram

- **Subject File:** Apart from traditional input/output parameters, each model may — during its execution — use parameters in the form of files. The files themselves are stored in the application's internal file storage system under the folder {rootFolderPath}/anonymizedPatients/{anonymizedPatientUuid}/{subjectUuid}.
  The pertinent information is stored in an intermediate linking table between Subject and Parameter, since a file is considered a special type of parameter, as described in the Model Repository. Although there is no direct file-upload mechanism associated with a specific subject, this data is stored in this table due to special care being taken to ensure the possibility of future downloading these files if it is ever required for further evaluation of a specific execution.

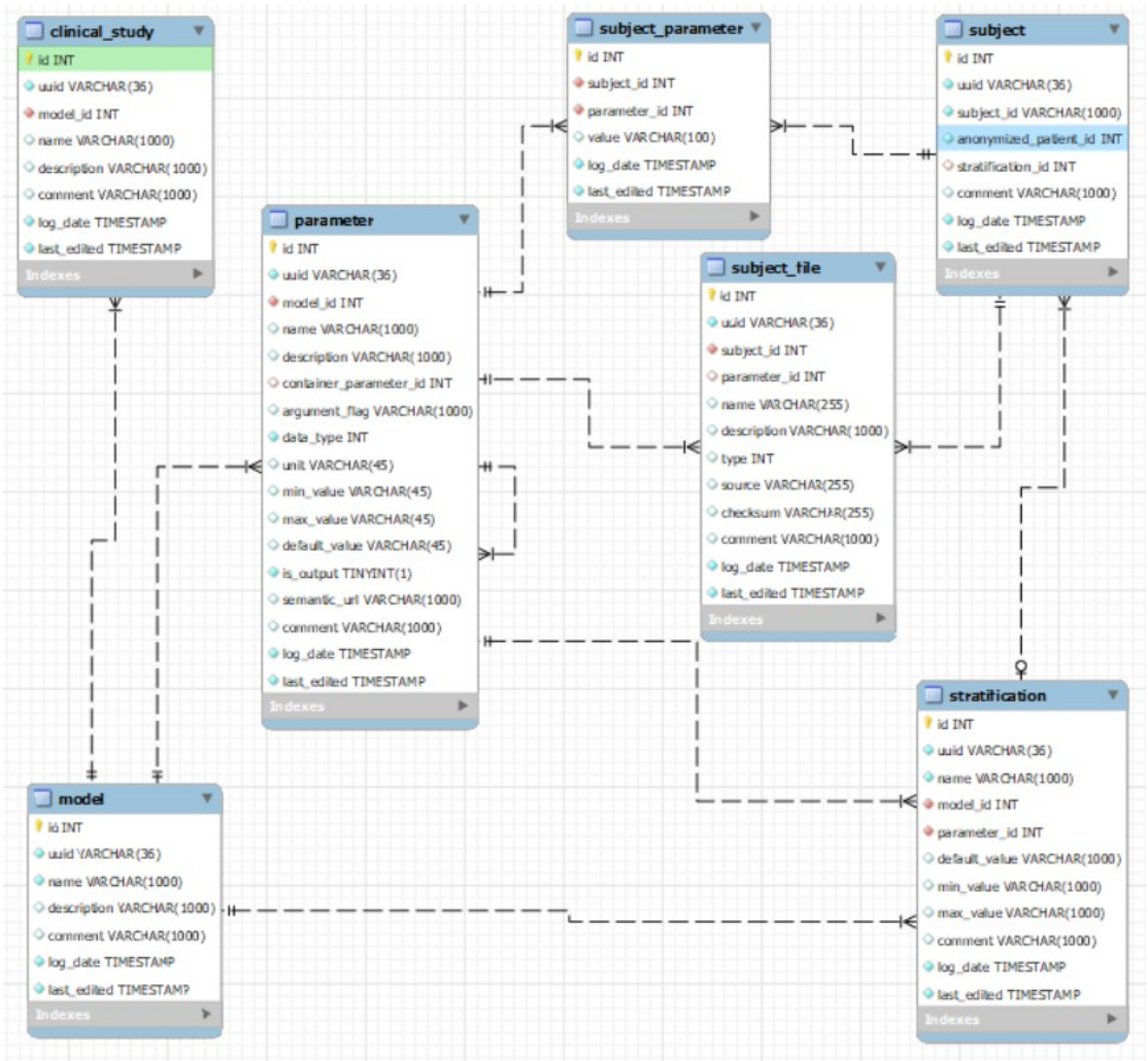


Figure 6-4: Clinical study – model repository coupling ER diagram

6.4. Clinical Decision Support Repository

The Clinical Decision Support Repository is based on the use of tables structured around the concept of a clinical question. Its purpose, according to the theoretical analysis in

Chapter 5, is to facilitate the easy and direct entry and management of clinical questions within the system, as well as their connections with simulation models and, by extension, their executions. This enables the clinician to make faster, more focused, and effective clinical decisions which will be tailored to the patient's initial condition and personalized according to their specific characteristics. The basic entity around which the repository is structured, as also depicted in the ER diagram of Figure 6-5, is the clinical question. The main principles governing this schema are as follows:

- Each clinical question has one or more clinical answers.
- Each clinical question can be related to one or more models.
- Each clinical question can be associated with a model through only one specific parameter of that model.
- An answer to a clinical question is associated with a specific value or a range of values that the question's related parameter can take.
- During the execution of a model, via the answer – parameter value association, the output parameters can be mapped to the appropriate answers. These answers are stored withing the model execution data and are used for the corresponding report.

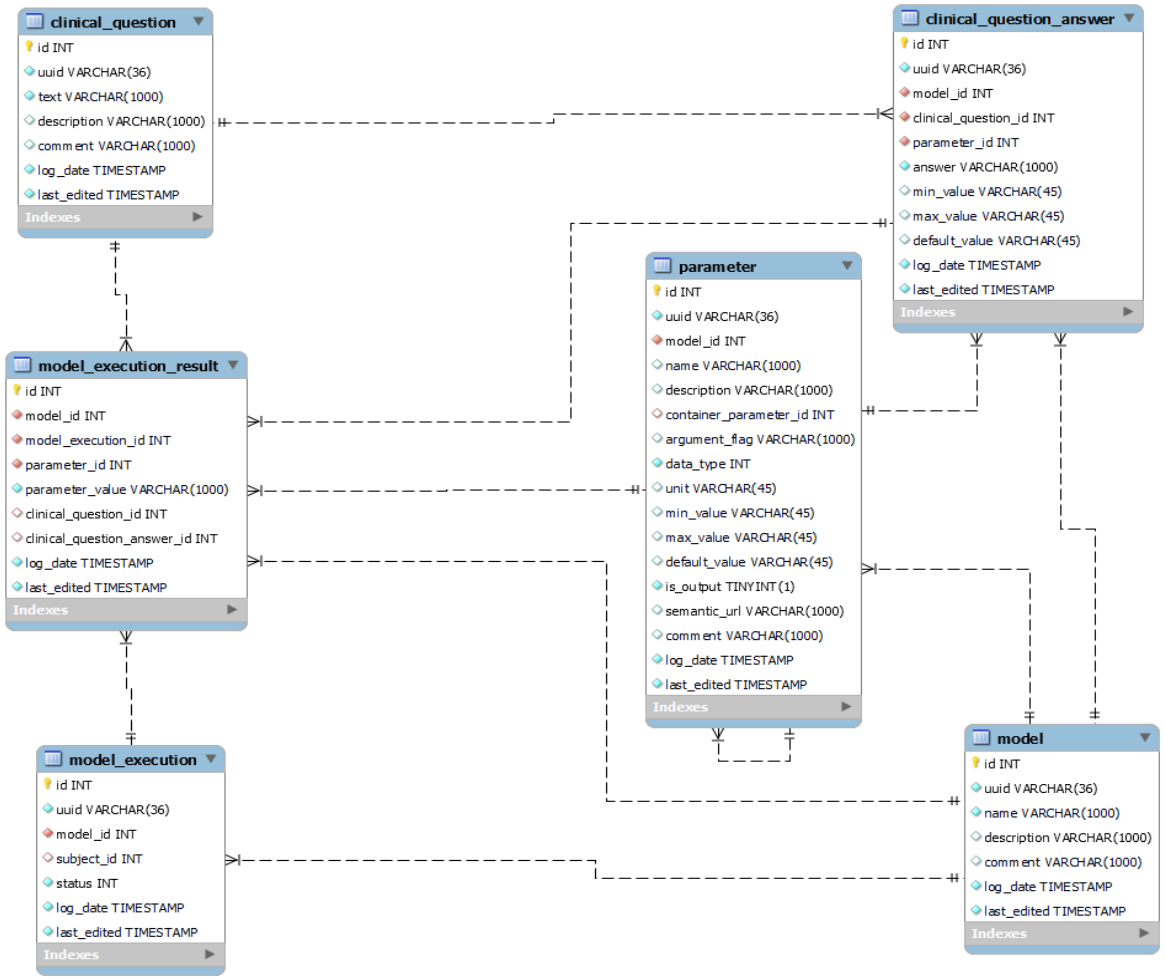


Figure 6-5: Clinical decision support repository ER diagram

The tables of the Clinical Decision Support Repository are as follows:

- **Clinical Question:** This is the basic entity of this repository, which forms the core of a Clinical Decision Support System (CDSS) operation. It contains the textual data of the question, such as the actual text and/or any additional description/comments.
- **Clinical Question Answer:** For each clinical question, there are corresponding answers related to both the model and parameter table of the Model Repository. This table contains the textual representation of the answer and the parameter values for which it is relevant.

- **Model Execution:** This is the second most important entity of the Clinical Decision Support Repository. It records model executions and links them to clinical questions thus allowing for the evaluation of individual cases, as well as the collection of data for analyses and reaching meaningful conclusions within the context of a Clinical Study. It should be noted here that the Model Execution table connects the Clinical Support Repository with the Clinical Study Repository through the Subject ID field, which refers to the Subject table, a necessary feature of a clinical study.
- **Model Execution Result:** This table connects the Model Execution and Clinical Question tables, functioning as a many-to-many relationship. The table is crucial because for each model execution it records the values of both the input and output parameters, along with the related clinical questions and their answers based on the aforementioned values. In this way, the data from each model execution are fully recorded, whether they originate from a specific Subject or from direct data entry via the user interface (cases of individual executions).

6.6. References

1. Evans, E. (2004). *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Addison-Wesley.

2. MySQL: https://www.mysql.com/

3. MyBatis: https://mybatis.org/mybatis-3/

4. Hibernate: https://hibernate.org/

5. Flyway: https://flywaydb.org/

6. Spring Boot: https://spring.io/projects/spring-boot

7. Java: https://www.oracle.com/java/

8. React: https://react.dev/

9. Stamatakos G. CHIC project, Deliverable D6.1: Cancer hypomodelling and hypermodelling strategies and initial component models

7. Results

In this chapter the individual functionalities of Hyperion will be presented. This presentation will be comprised of two parts. First, the crud functionality for each repository entity will be explained step by step. In the second part, specialized workflows regarding the execution of a model, the handling of the execution results and the set-up and management of a clinical study will be demonstrated.

7.1. Basic CRUD workflows

7.1.1 Model Repository

7.1.1.1 Model

To manage the stored models a user must click on the "Models" option from the sidebar. This will take them to the list of model descriptions. By clicking the "Add Model" button on the top left side of the screen, they will be navigated to the model creation screen. In the presented form, they can enter the descriptive information for the new model. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the model list screen, where they can view their new entry.

Editing a model is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the model values and then save the changes to the repository by clicking the “Submit” button.

Deleting a model is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the model is deleted from the database and the list screen is reloaded.

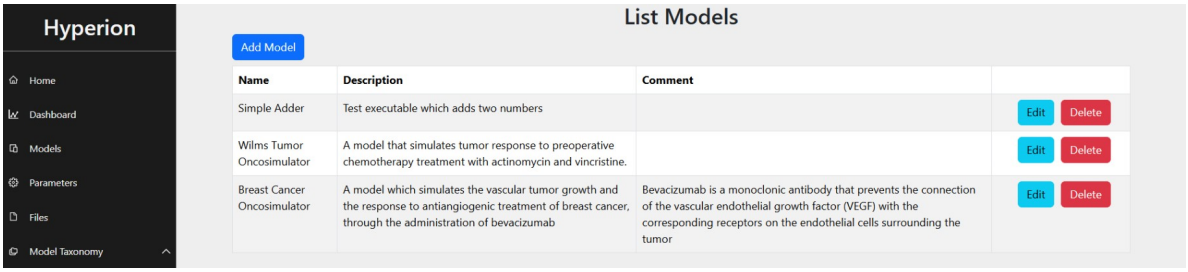


Figure 7-1: Model list screen

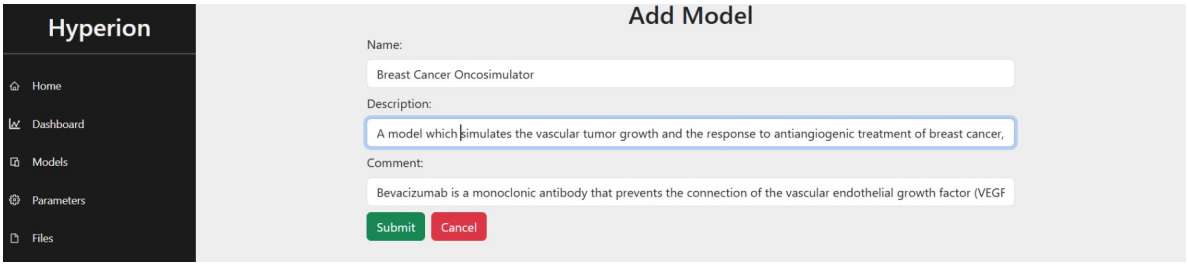


Figure 7-2: Add/Edit model screen. For editing, an extra non-editable field is provided showing the UUID of the model which is automatically assigned at the time of input.

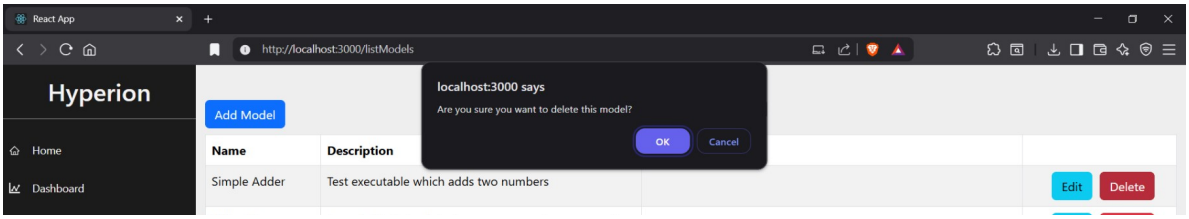


Figure 7-3: Model deletion confirmation pop-up. Similar popups are being produced for all listed entities across Hyperion.

7.1.1.2 Parameter

To manage the stored parameters a user must click on the "Parameters" option from the sidebar. This will take them to the list of parameters. In that screen, the user can see in a paginated table all the stored parameters. They can also use the drop-down at the top and choose a model to filter the page content and view the parameters of the chosen model.

By clicking the "Add Parameter" button on the top left side of the screen, they will be navigated to the parameter creation screen. In the presented form, they can enter the information for the new parameter. Among the input fields, there is a drop-down field containing a list of all stored models, which is requested upon loading the screen. It is mandatory for the user to choose a model, since parameters are an integral part of a model, as shown in the ER diagram in chapter 6.2. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the parameter list screen, where they can view their new entry.

Editing a parameter is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the parameter values and then save the changes to the repository by clicking the “Submit” button.

Deleting a parameter is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the parameter is deleted from the database and the list screen is reloaded.

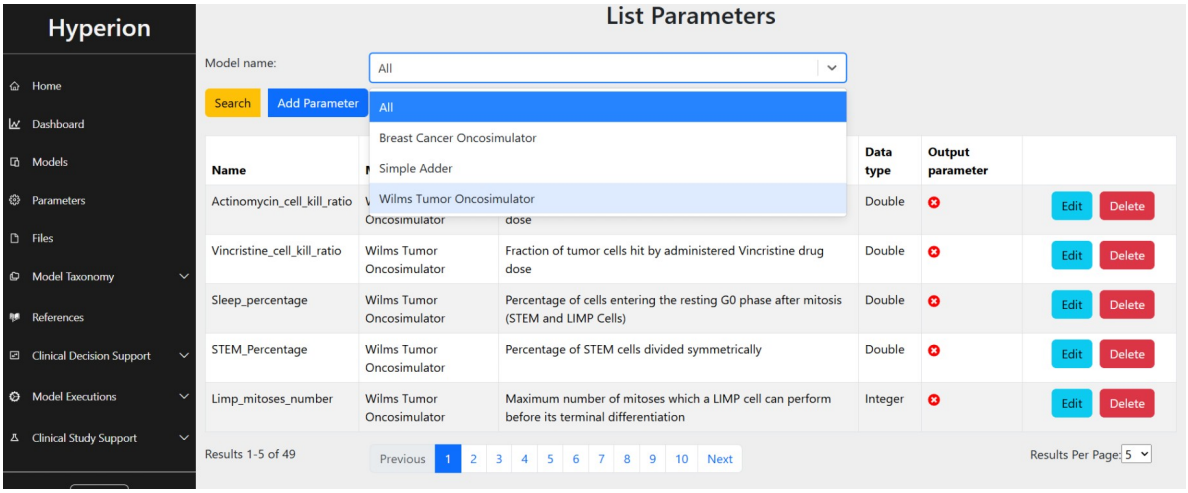


Figure 7-4: Parameter list screen with pagination and model-based filtering

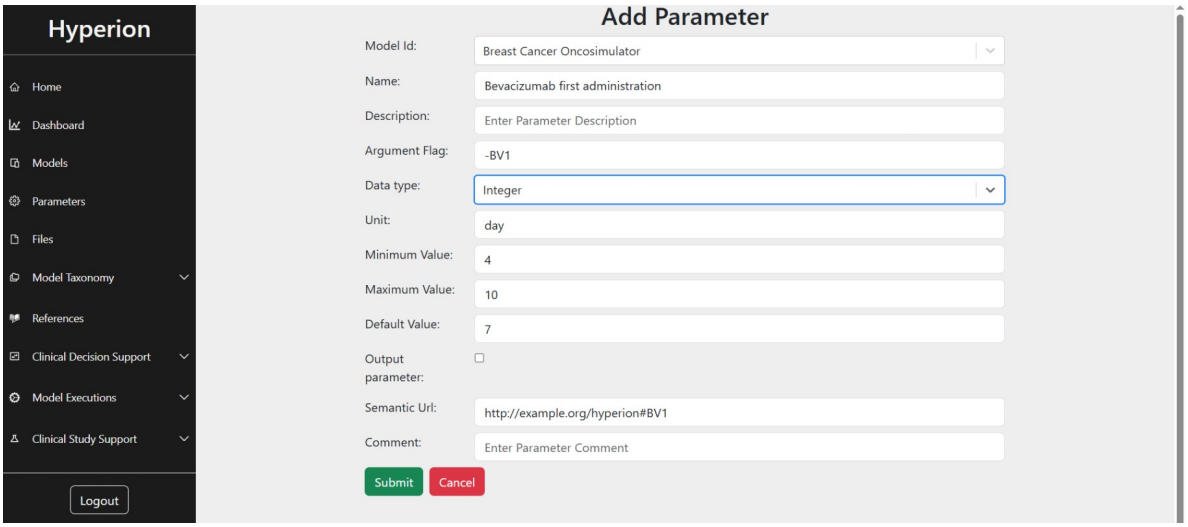


Figure 7-5: Add/Edit parameter screen. For editing, an extra non-editable field is provided showing the UUID of the parameter which is automatically assigned at the time of input.

7.1.1.3 File

To manage the stored files a user must click on the "Files" option from the sidebar. This will take them to the list of files. In that screen, the user can see in a paginated table all the stored files. They can also use the drop-down at the top and choose a model to filter the page content and view the files of the chosen model.

By clicking the "Add File" button on the top left side of the screen, they will be navigated to the file creation screen. In the presented form, they can enter the information for the new file. Among the input fields, there is a drop-down field containing the list of all stored models, which is requested upon loading the screen. It is mandatory for the user to choose a model, since files are an integral part of a model, as shown in the ER diagram in chapter 6.2, be it either the model executable itself, a text guide, or a supplementary script. In that same screen, the user is required to click on the “Choose file” button of the “Source” field. This will result in a file choosing pop-up from the OS. The user must choose a file and click “Ok”. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the file list screen, where they can view their new entry. The file itself will be uploaded to Hyperion’s file storage system under the folder {rootFolderPath}/models/{modelUuid}.

Editing a file is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the file values and then save the changes to the repository by clicking the “Submit” button. It should be noted that in that screen, the file itself can be downloaded and only some descriptive information can be edited.

Deleting a file is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the file info is deleted from the database, the file itself is deleted from the file storage and the list screen is reloaded.

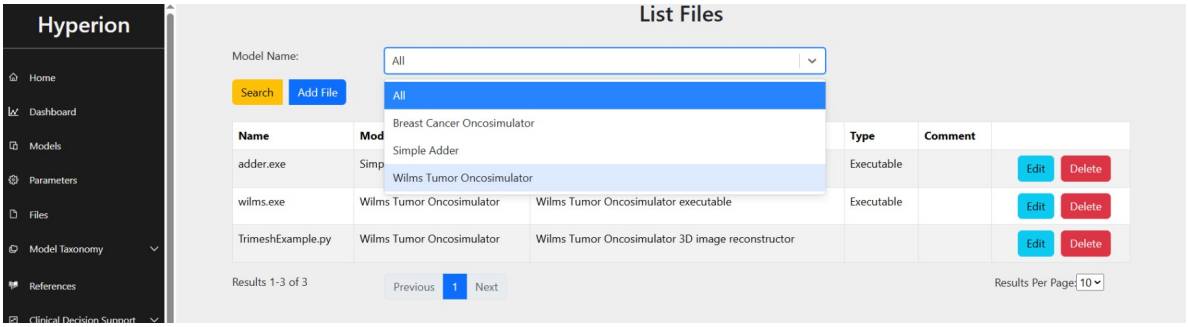


Figure 7-6: File list screen with pagination and model-based filtering

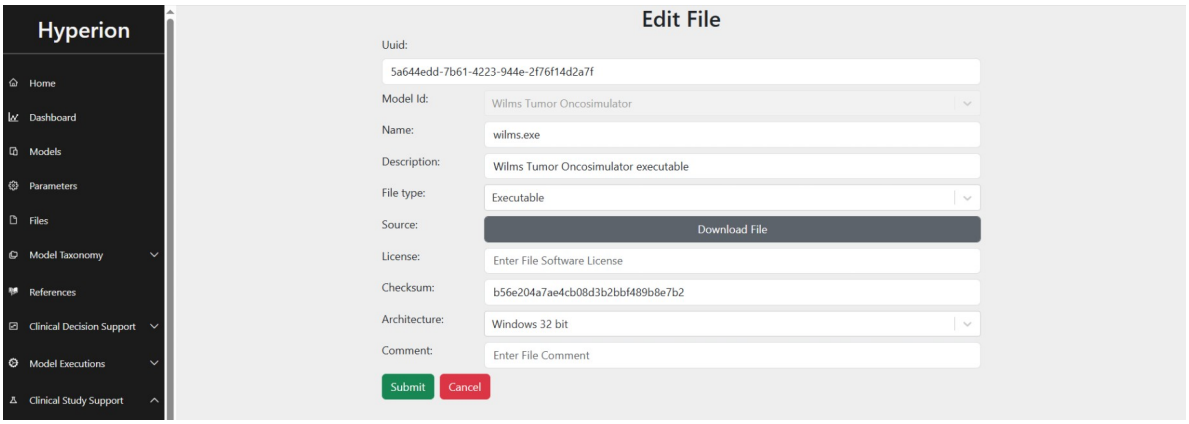


Figure 7-7: Edit/download file screen

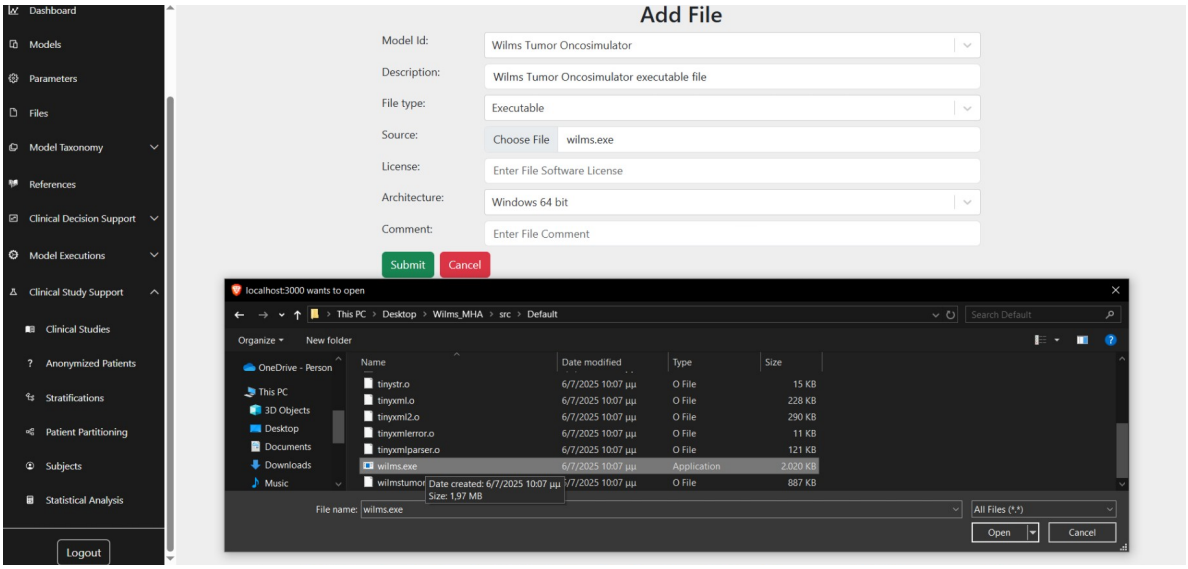


Figure 7-8: Add file screen, with the OS browser open to choose a file to upload

7.1.1.4 Property

To manage the stored properties a user must click on the "Model Taxonomy → Properties" option from the sidebar. This will take them to the list of properties. In that screen, the user can see all the stored properties.

By clicking the "Add Property" button on the top left side of the screen, they will be navigated to the property creation screen. In the presented form, they can enter the information for the new property. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the parameter list screen, where they can view their new entry.

Editing a property is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the property values and then save the changes to the repository by clicking the “Submit” button.

Deleting a property is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the property is deleted from the database and the list screen is reloaded. Due to the model-property many-to-many relationship, if a property is deleted, then the corresponding rows from the intermediate table will also be deleted, as the pertinent foreign keys are set to CASCADE mode.

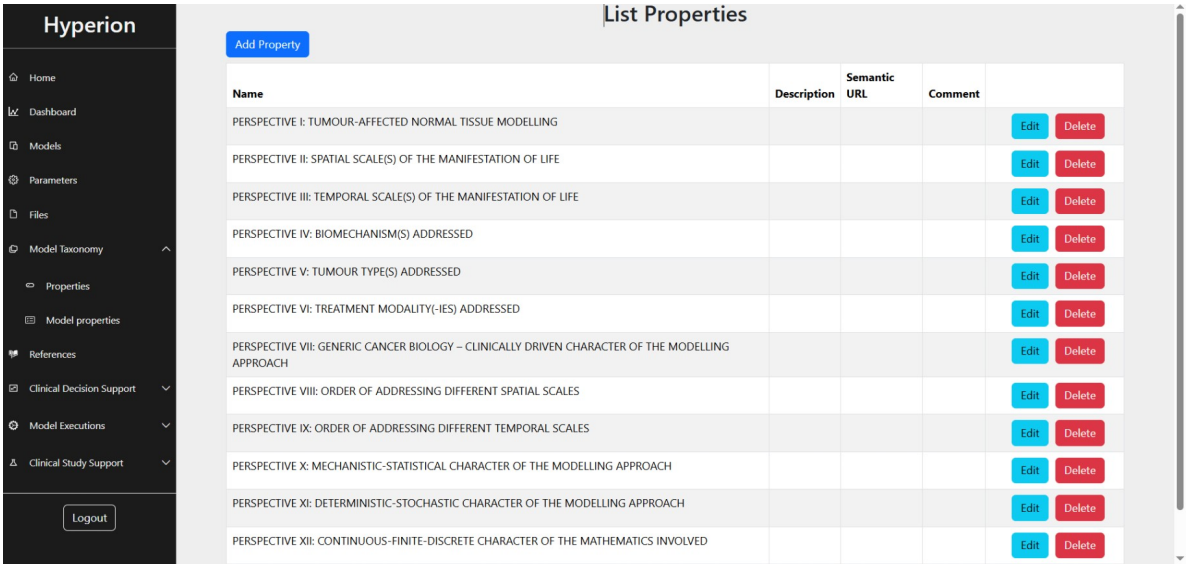


Figure 7-9: Property list screen

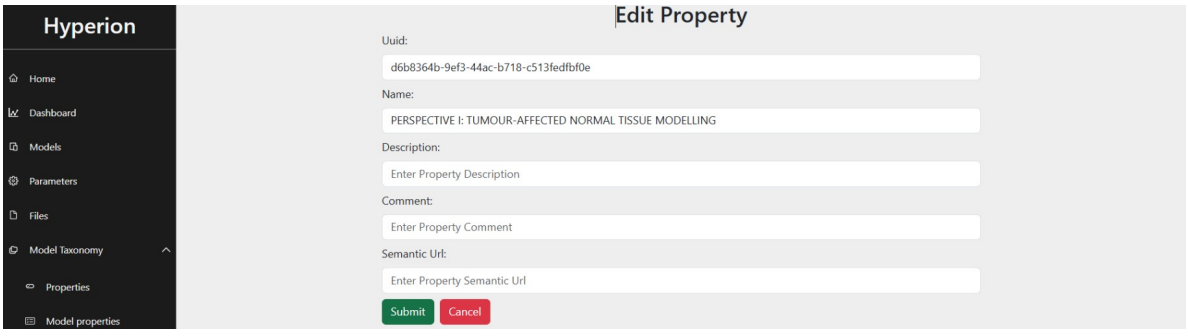


Figure 7-10: Add/Edit property screen. For editing, an extra non-editable field is provided showing the UUID of the property which is automatically assigned at the time of input.

7.1.1.5 Model – Property combination

To manage the stored model – property combinations a user must click on the "Model Taxonomy → Model Properties" option from the sidebar. This will take them to the list of combinations. In that screen, the user can see all the stored combinations.

By clicking the "Add Property values for a Model" button on the top left side of the screen, they will be navigated to the combination creation screen. In the presented form, they must first choose the model of the combination from the drop-down on the top of the screen. Then they can enter the values for as many properties as they want. Each property-related set of fields contains an “Add” and a “Remove” button. The user can therefore manage the number of properties they want to add values for. Since each model – property combination can only have one value, each time a new field combination of fields is added for a property, the drop-down list of available properties is updated to contain only the non-used properties so far. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the model – property list screen, where they can view their new entries.

Editing a model – property combination is possible from the list screen, by clicking the “Edit” button on the right side of its row. Unlike the add screen, this screen only edits the chosen combination. The property drop-down field is set up to contain only the available properties that the model can be related to. In the case that the model of the combination remains, then the available properties are comprised of the property of the combination, plus all the non-related ones. In case another model is chosen, then only the non-related properties are returned.

Deleting a model – property combination is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the combination is deleted from the database and the list screen is reloaded.

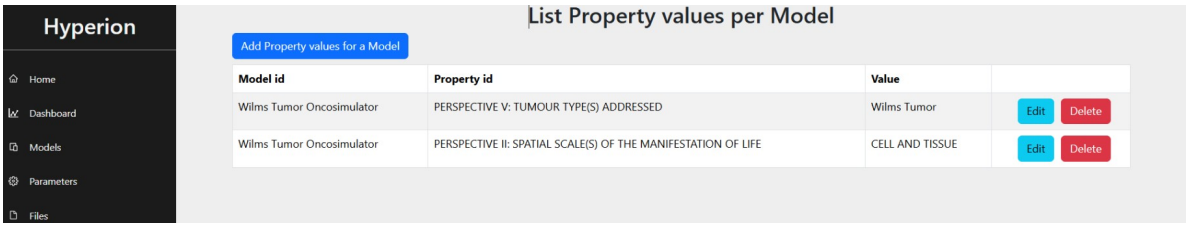


Figure 7-11: Model – property combination list screen

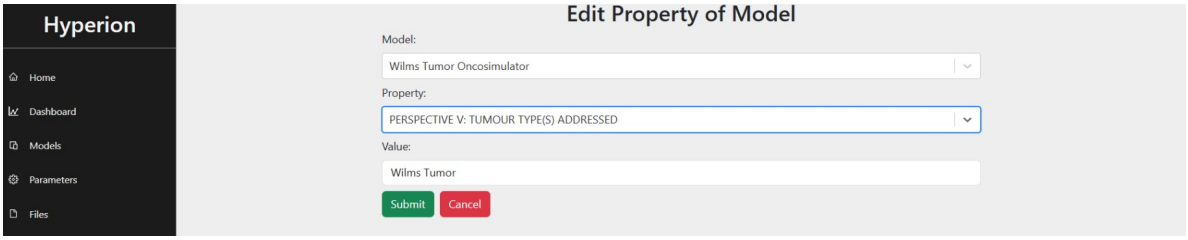


Figure 7-12: Model – property combination edit screen

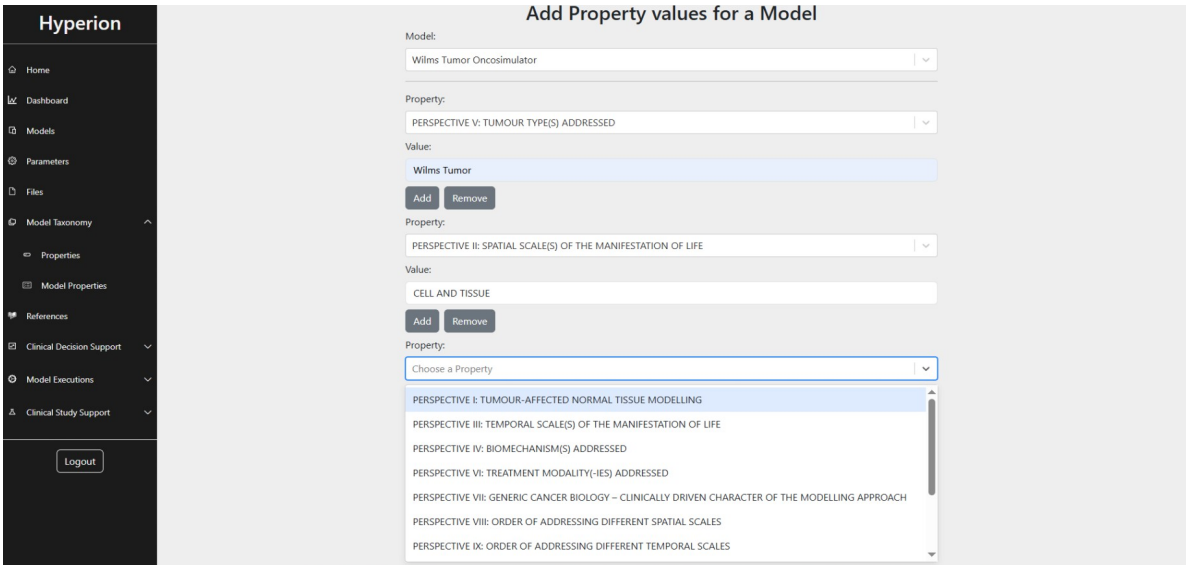


Figure 7-13: Add model-property combinations screen.

7.1.1.6 Reference

To manage the stored references a user must click on the "References" option from the sidebar. This will take them to the list of references. By clicking the "Add Reference" button on the top left side of the screen, they will be navigated to the reference creation screen. In the presented form, they can enter the information for the new reference. Among the input fields, there is a drop-down field containing the list of all stored models, which is requested upon loading the screen. It is mandatory for the user to choose a model, since references are related to a model, as shown in the ER diagram in the previous chapter. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the reference list screen, where they can view their new entry.

Editing a reference is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the reference values and then save the changes to the repository by clicking the “Submit” button.

Deleting a reference is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the reference is deleted from the database and the list screen is reloaded.

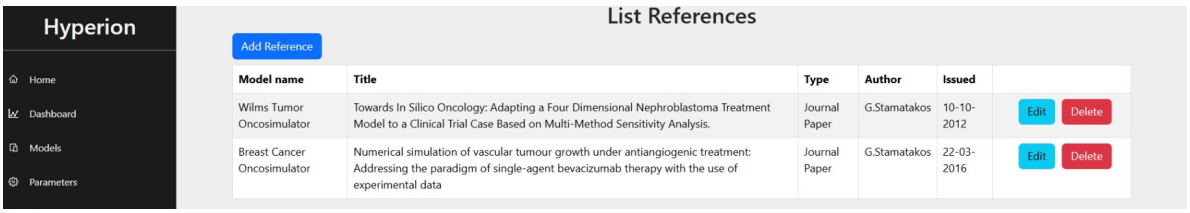


Figure 7-14: Reference list screen

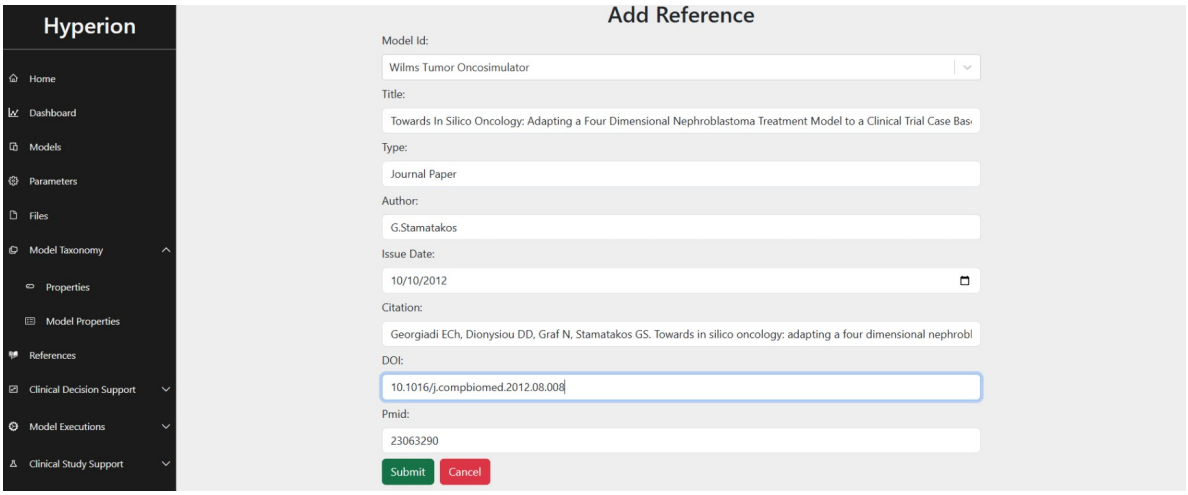


Figure 7-15: Add/Edit reference screen. For editing, an extra non-editable field is provided showing the UUID of the reference which is automatically assigned at the time of input.

7.1.2. Clinical Study Repository

7.1.2.1. Clinical Study

To manage the stored clinical studies a user must click on the "Clinical Study Support → Clinical Studies" option from the sidebar. This will take them to the list of clinical . In that screen, the user can see in a paginated table all the stored parameters.

By clicking the "Add Clinical Study" button on the top left side of the screen, they will be navigated to the clinical study creation screen. In the presented form, they can enter the information for the new clinical study. Among the input fields, there is a drop-down field containing all stored models, which is requested upon loading the screen. It is mandatory for the user to choose a model, since clinical studies are conducted using models, which is shown in the ER diagram in chapter 6.3. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the clinical study list screen, where they can view their new entry.

Editing a clinical study is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the clinical study values, except for the related model and then save the changes to the repository by clicking the “Submit” button.

Deleting a clinical study is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the clinical study is deleted from the database and the list screen is reloaded.

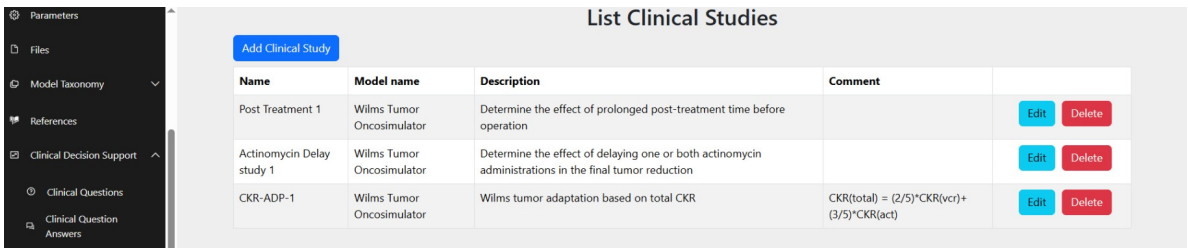


Figure 7-16: Clinical Study list screen

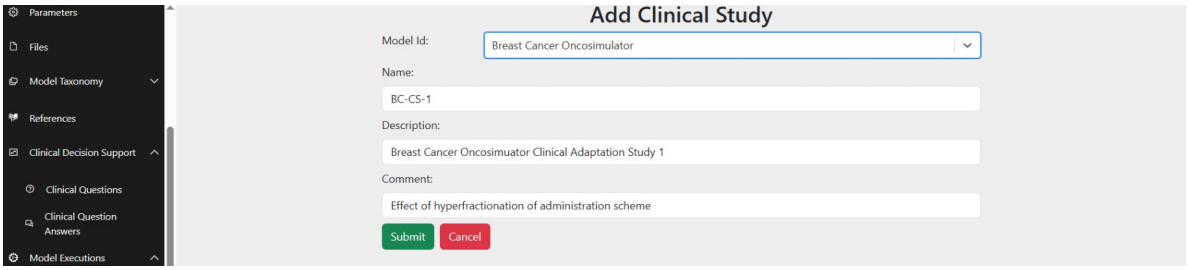


Figure 7-17: Add Clinical Study screen

Figure 7-18: Edit Clinical Study screen

7.1.2.2. Anonymized Patients

To manage the stored anonymized patients a user must click on the "Clinical Study Support → Anonymized Patients" option from the sidebar. This will take them to the list of anonymized patients. In that screen, the user can see in a paginated table all the stored patients. They can also use the drop-down at the top and choose a clinical study to filter the page content and view the patients of the chosen clinical study.

By clicking the "Add Patient" button on the top left side of the screen, they will be navigated to the patient creation screen. In the presented form, they can enter the information for the new patient. Among the input fields, there is a drop-down field containing all clinical studies, which is requested upon loading the screen. It is mandatory for the user to choose a study, since anonymized patients are only used in the context of clinical studies, as shown in the ER diagram in chapter 6.3. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the patient list screen, where they can view their new entry.

Editing a patient is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the patient values and then save the changes to the repository by clicking the “Submit” button.

Deleting a patient is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the patient is deleted from the database and the list screen is reloaded.

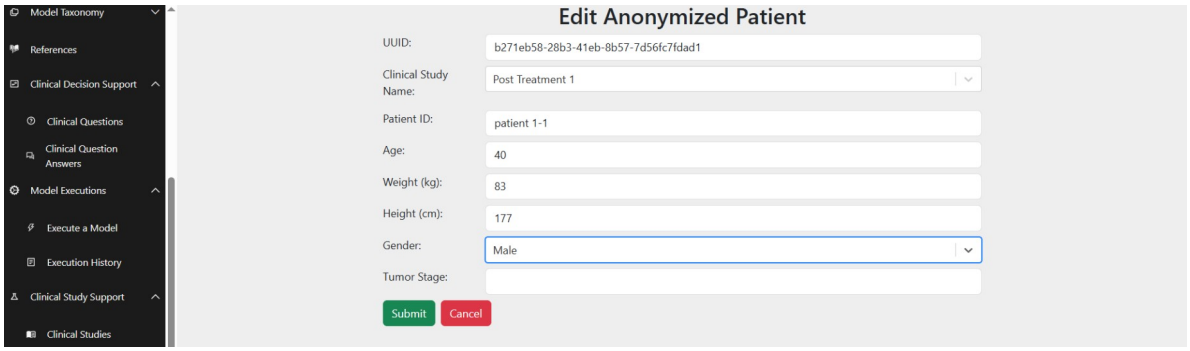


Figure 7-19: Add/Edit anonymized patient screen. For editing, an extra non-editable field is provided showing the UUID of the anonymized patient which is automatically assigned at the time of input.

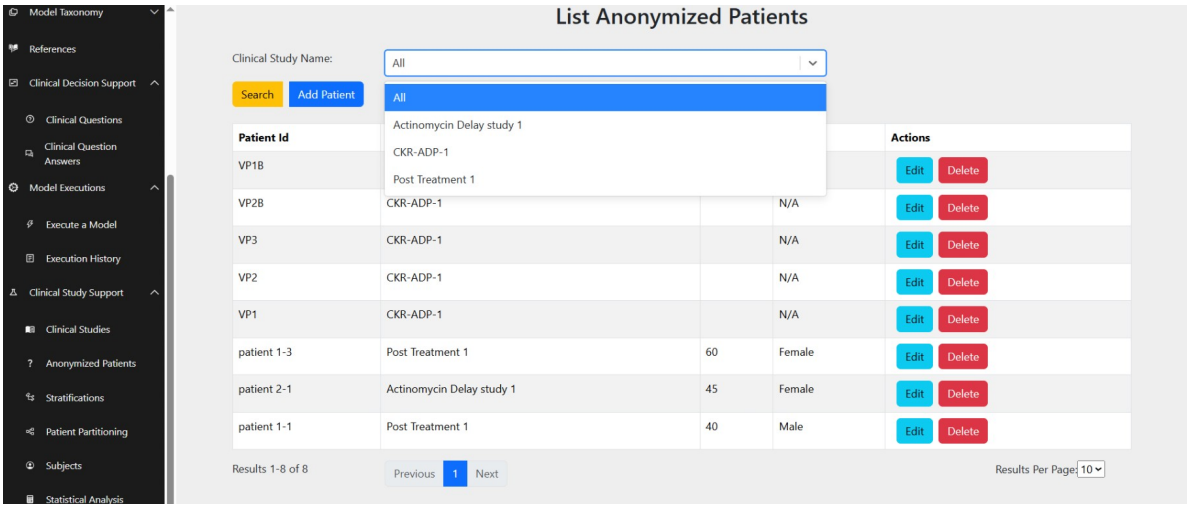


Figure 7-20: Anonymized patient list screen

7.1.2.3. Stratification

To manage the stored stratifications a user must click on the "Clinical Study Support → Stratifications" option from the sidebar. This will take them to the list of anonymized patients. In that screen, the user can see in a paginated table all the stored stratifications. There also two drop-downs to filter the content. First a model can be chosen, so that only the stratifications of that model are shown. Upon choosing a model, the parameter drop-down gets populated with the model's parameters, adding a second filter for the stratifications.

By clicking the "Add Stratification" button on the top left side of the screen, they will be navigated to the stratification creation screen. In the presented form, they can enter the information for the new patient. Taking into account the ER diagram in chapter 6.3, the creation screen has two mandatory drop-down fields pertaining to the model and the parameter of the stratification. These two drop-downs work in the same sequential way as in the list screen. It is also mandatory that a single value or a value range for the chosen parameter is given. After clicking the "Submit" button, the data will be stored in the

database, and the user will be redirected back to the stratification list screen, where they can view their new entry.

Editing a stratification is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the stratification values and then save the changes to the repository by clicking the “Submit” button.

Deleting a stratification is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the patient is deleted from the database and the list screen is reloaded.

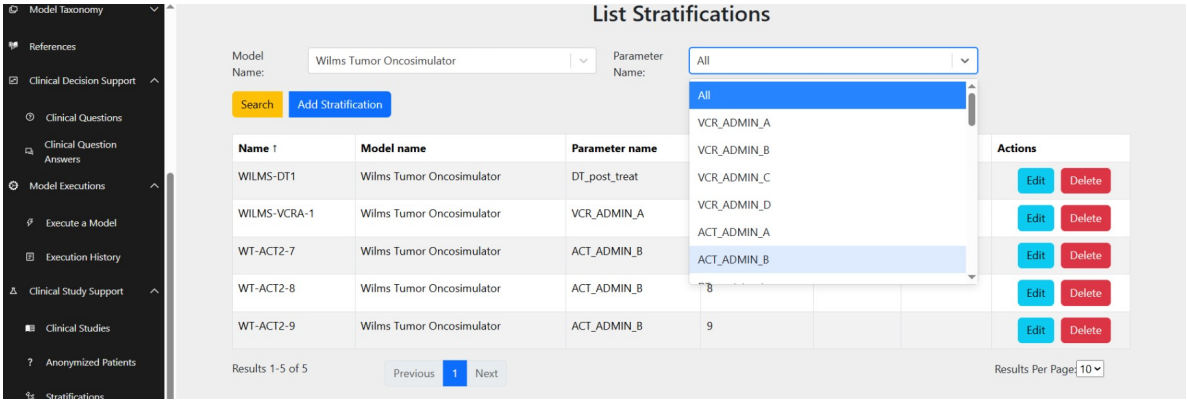


Figure 7-21: Stratification list screen

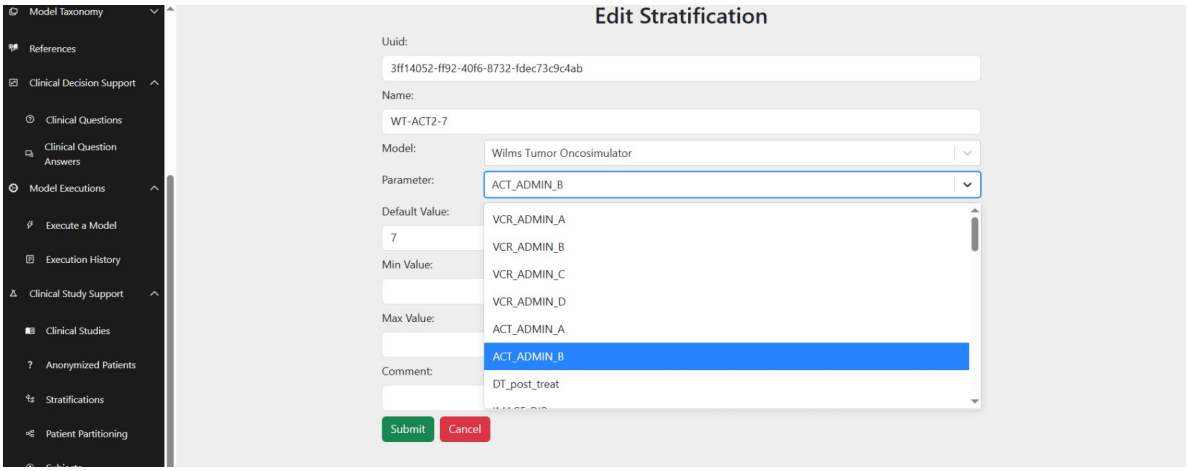


Figure 7-22: Add/Edit stratification screen. For editing, an extra non-editable field is provided showing the UUID of the stratification which is automatically assigned at the time of input.

7.1.2.4. Patient Partitioning

Patient partitioning involves creating combinations between anonymized patients and stratifications, which share a many-to-many relation. To manage these combinations a user must click on the "Clinical Study Support → Patient Partitioning" option from the sidebar. This will take them to the list of combinations where the user can see in a paginated

table all the stored combinations. There also two drop-downs to filter the content based on either the anonymized patients, or the stratifications.

By clicking the "Add an Anonymous Patient to a Stratification" button on the top left side of the screen, they will be navigated to the combination creation screen. In the presented form, they must first choose the anonymized patient of the combination from the drop-down on the top of the screen. Then they can enter the values for as many stratifications as they want. Each stratification-related set of fields contains an “Add” and a “Remove” button. The user can therefore manage the number of stratifications that they want to add. Since each patient can only participate one in a stratification, each time a new field combination of fields is added for a property, the drop-down list of available stratifications is updated to contain only the non-used entities so far. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the combination list screen, where they can view their new entries.

Editing an anonymized patient – stratification is possible from the list screen, by clicking the “Edit” button on the right side of its row. Unlike the add screen, this screen only edits the chosen combination. The stratification drop-down field is set up to contain only the available properties that the patient can be related to. In the case that the patient of the combination remains, then the available stratifications are comprised of the property of the combination, plus all the non-related ones. In case another patient is chosen, then only the non-related stratification are returned.

Deleting an anonymized patient – stratification combination is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the combination is deleted from the database and the list screen is reloaded.

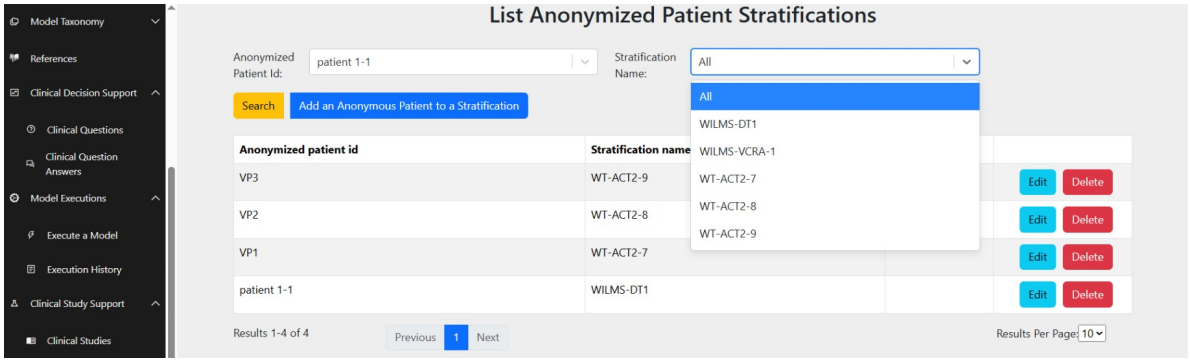


Figure 7-23: Anonymized patient – stratification combination list screen

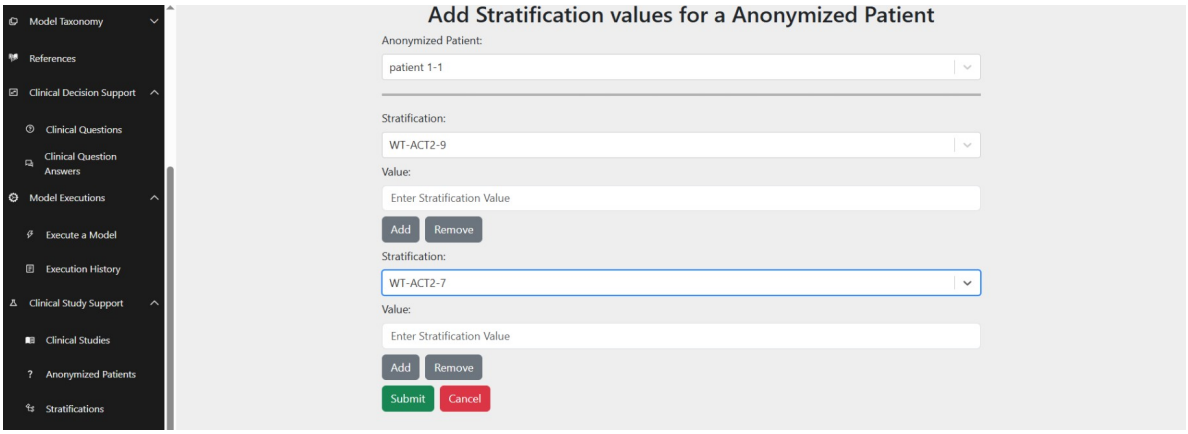


Figure 7-24: Add anonymized patient – stratification combinations screen

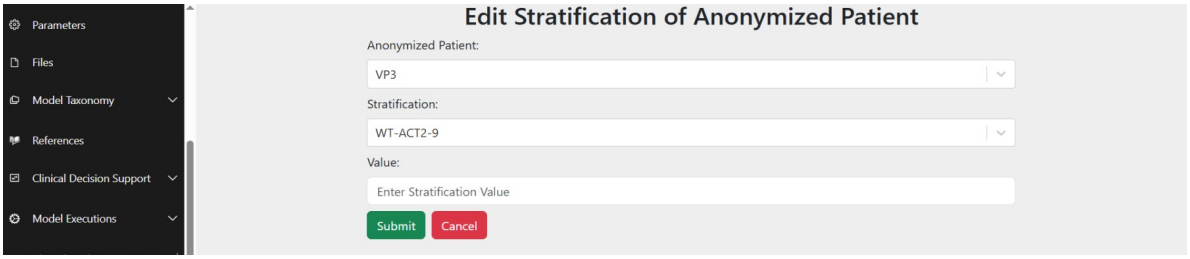


Figure 7-25: Edit anonymized patient – stratification combination screen

7.1.3.5. Subject

To manage the stored subjects a user must click on the "Clinical Study Support → Subjects" option from the sidebar. This will take them to the list of subjects. In that screen, the user can see in a paginated table all the stored subjects. They can also use the drop-downs at the top as filters and view the subjects related either to an anonymized patient or a stratification

By clicking the "Add Subject" button on the top left side of the screen, they will be navigated to the subject creation screen. In the presented form, they can enter the information for the new subject. When the screen is initially loaded, the user has to choose the anonymized patient that the subject will be related to. Once that is done, the form will be updated with the id of the clinical study that the anonymized patient is part of, and the id of the study's related model. Due to that, fields corresponding to the model inputs will be shown to the user, and the stratification drop-down will be populated with the proper stratifications that are available via the chosen model and its parameters. In addition, once a stratification is chosen, if it is defined using a default parameter value, that value will automatically populate the corresponding input field, which will be disable for editing. After this procedure, the user can fill in the rest of the input variable fields. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the subject list screen, where they can view their new entry.

Editing a subject is possible from the list screen, by clicking the "Edit" button on the right side of its row. This leads to the edit screen, where the user can change only the input

values of the model parameters or the stratification. In the case of the latter, the same mechanism which will hardcode an input value as in the creation screen will be applied. Then user can then save the changes to the repository by clicking the “Submit” button.

Deleting a subject is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the subject is deleted from the database and the list screen is reloaded.

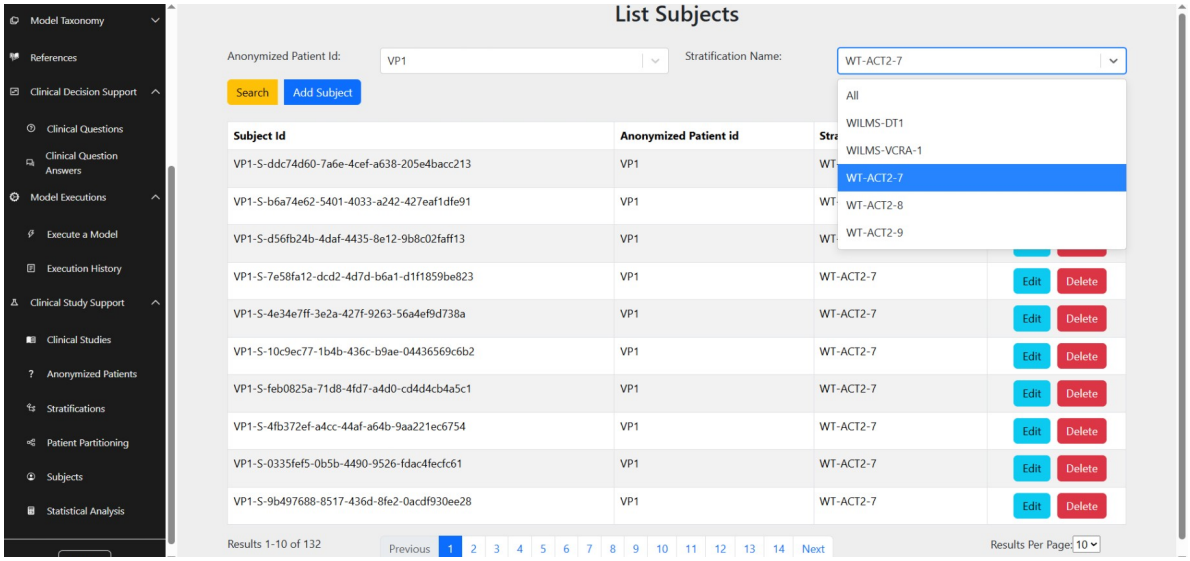


Figure 7-26: Subject list screen

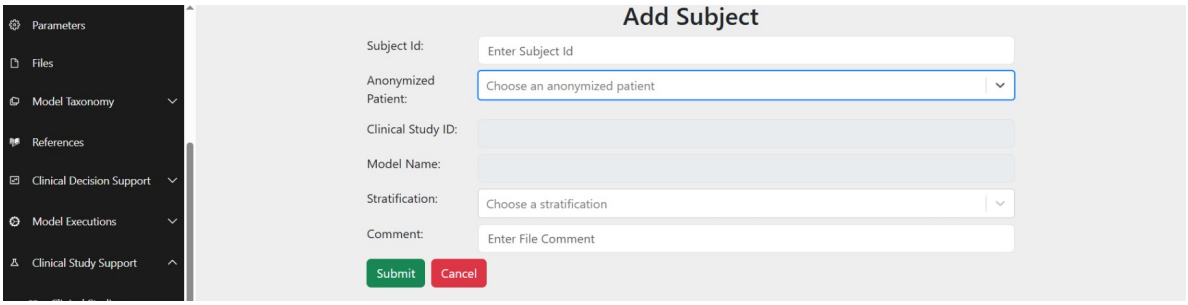


Figure 7-27: Add subject screen. Upon choosing an anonymized patient the screen will be updated with the proper input fields, resulting in a similar structure with the Edit screen

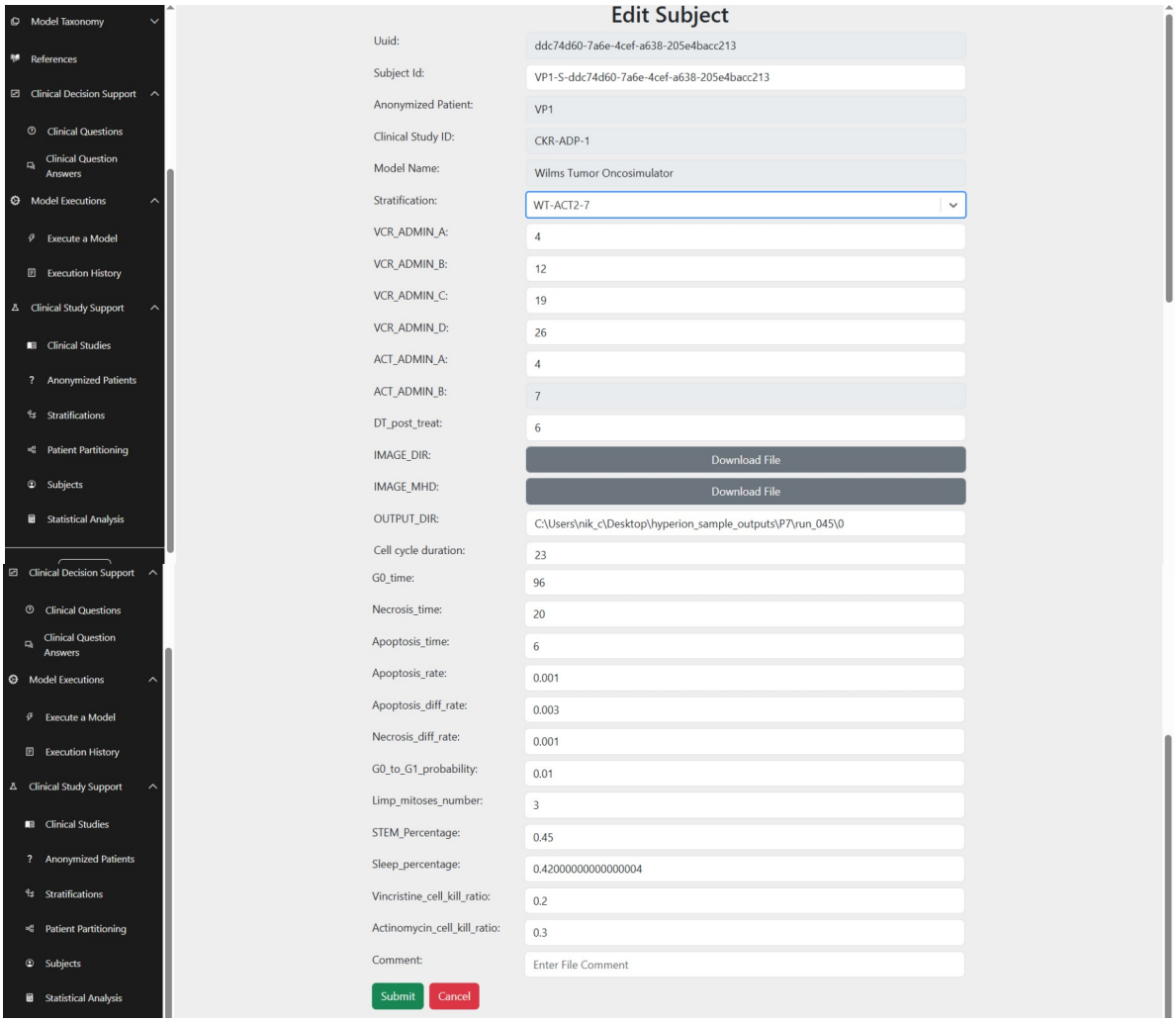


Figure 7-28: Edit subject screen

7.1.3. Clinical Decision Support Repository

7.1.3.1. Clinical Question

To manage the stored clinical questions a user must click on the "Clinical Decision Support → Clinical Questions" option from the sidebar. This will take them to the list of model descriptions. By clicking the "Add Clinical Question" button on the top left side of the screen, they will be navigated to the clinical question creation screen. In the presented form, they can enter the basic textual information for the new clinical question. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the clinical question list screen, where they can view their new entry.

Editing a clinical question is possible from the list screen, by clicking the “Edit” button on the right side of its row. This leads to the edit screen, where the user can change the clinical question values and then save the changes to the repository by clicking the “Submit” button.

Deleting a clinical question is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the clinical question is deleted from the database and the list screen is reloaded.

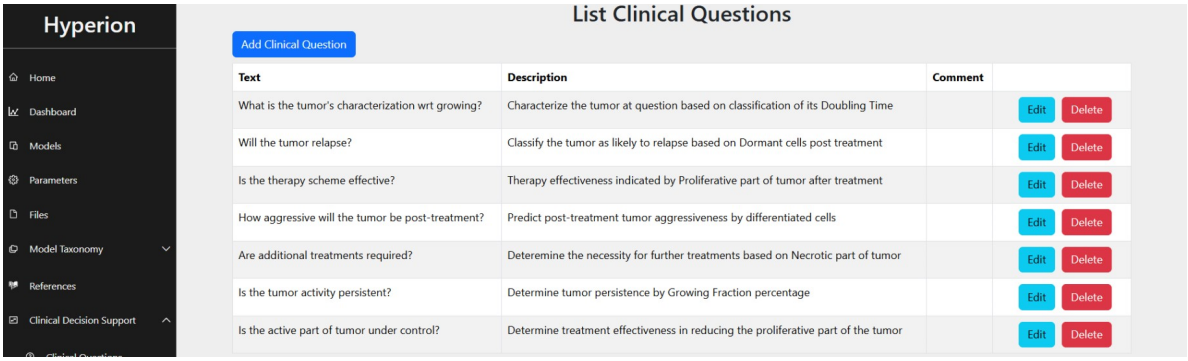


Figure 7-29: Clinical question list screen

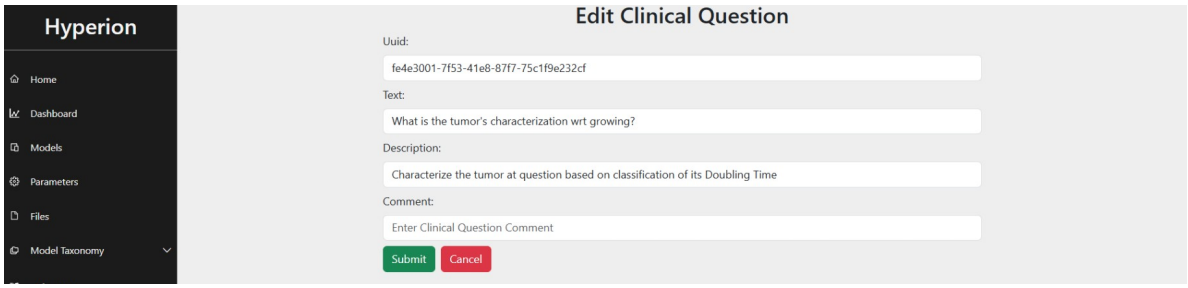


Figure 7-30: Add/Edit clinical question screen. For editing, an extra non-editable field is provided showing the UUID of the clinical question which is automatically assigned at the time of input.

7.1.3.2. Clinical Question Answer

To manage the stored files a user must click on the "Clinical Decision Support → Clinical Question Answers" option from the sidebar. This will take them to the list of answers. In that screen, the user can see in a paginated table all the stored answers. They can also use the drop-down at the top and choose a model to filter the page content and view the answers handled by the chosen model.

By clicking the "Add Answers to a Clinical Question" button on the top left side of the screen, they will be navigated to the answer creation screen. In the presented form, they must first choose the model that will be used to give the answer to the clinical question from the drop-down on the top of the screen. Then they can enter the values for as many questions-answer pairs as they want. For each pair a different question can be chosen each time, and its answer can be defined based on a certain parameter. Both choices are mandatory, as explained in chapter 6.4 Value-vise, it is also mandatory to give either a default value or a value range to define the answer. Each question-answer pair-related set of fields contains an “Add” and a “Remove” button. The user can therefore manage the number of pairs they want to add. After clicking the "Submit" button, the data will be stored in the database, and the user will be redirected back to the clinical question answer list screen, where they can view their new entries.

Editing an answer is possible from the list screen, by clicking the “Edit” button on the right side of its row. Unlike the add screen, this screen only edits the pair that is

represented by the chosen entry. However both the question, the answer and the values can be changed.

Deleting an answer is possible from the list screen, by clicking the “Delete” on the right side of the row, which will result in a confirmation pop-up. Once the user clicks “Ok” the answer is deleted from the database and the list screen is reloaded.

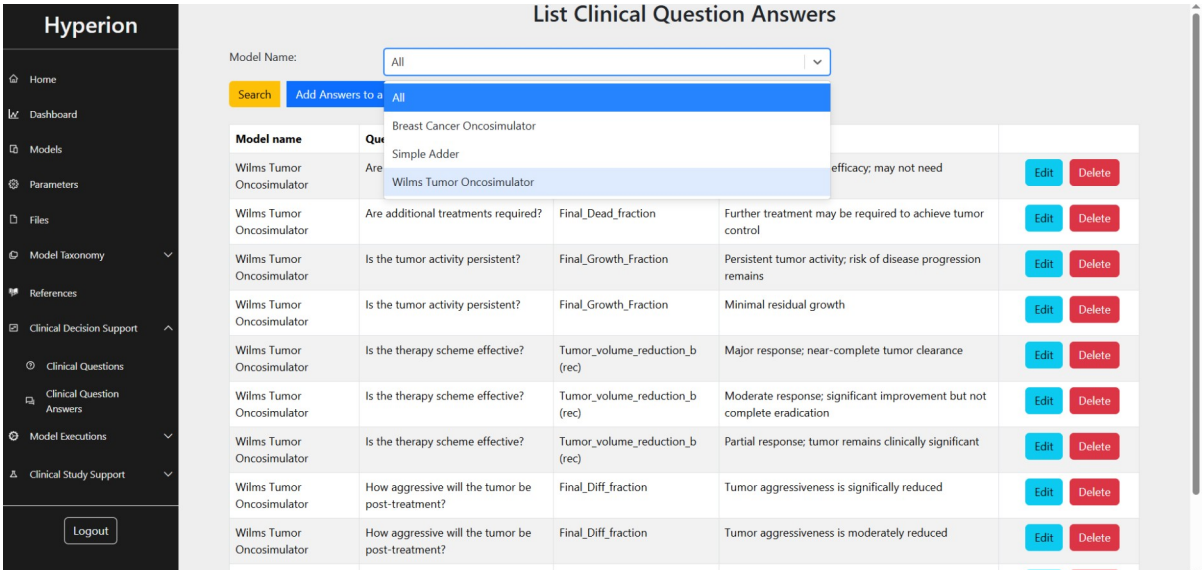


Figure 7-31: Clinical question answer list screen

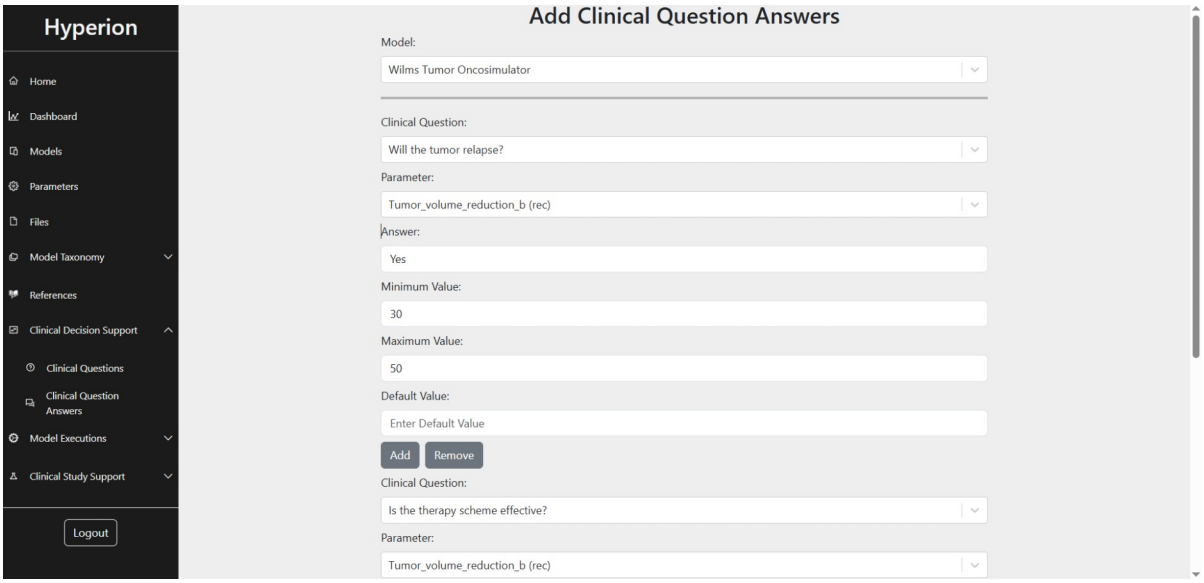


Figure 7-32: Add clinical question answer screen

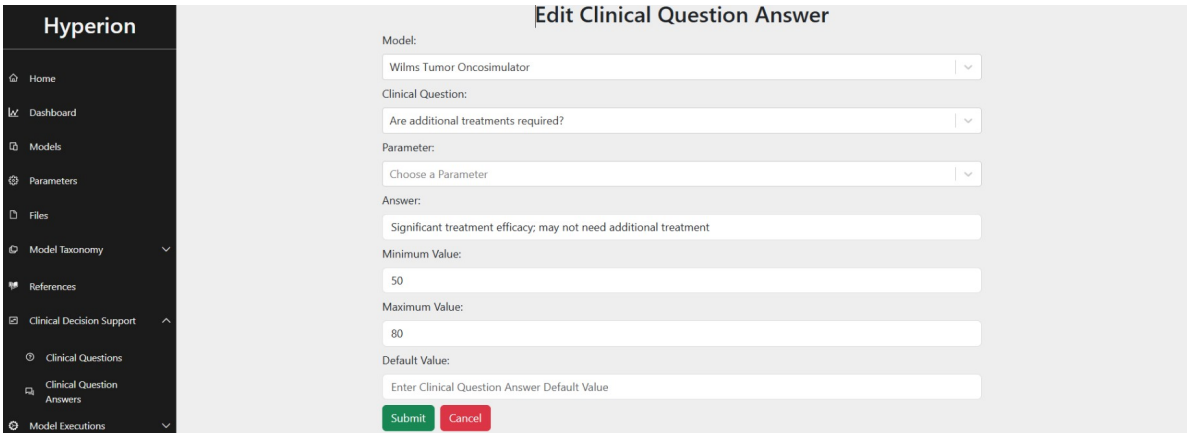


Figure 7-33: Edit clinical question answer screen.

7.2 Model execution management workflows

7.2.1 Model Execution and Reports

To execute a model, the user must click on the "Model Execution→Execute a model" option. This will lead them to the Model Execution Wizard. This wizard is comprised of three different pages

- Model Selection Page: The used chooses a model from the ones stored in the Model Repository.
- Data Input: After the first step the user is taken to the next screen where they will enter the input values for the model execution. To that end a call is made to the back end to retrieve the input parameters of the model. The call's response data are then used to construct the form's fields, in which the input values will be given.
  Alternatively, the user can choose a subject from the drop-down menu directly below the model name field. This drop-down will be active and populated with data if the model participated in a clinical study. In that case, the input fields will be automatically filled with the stored parameter values of the subject. Additionally, the field that holds the parameter which defines the subject's stratification will be disabled, as that value cannot be altered due to the stratification's definition, as described in chapter 6.4
- Data summary: Prior to the model execution, the user can view the values that will be sent to the back end to execute the model. Upon clicking "Next" the data will be submitted and the model execution will start.

The user has also access to the model execution reports. By clicking "Model Executions" and then "Execution History", the user will be presented with a list screen containing all the model executions sorted from the most recent to the oldest. The user can then click on the "View" button on the desired row, which will bring forth the model execution report screen. This screen, although different for each model based on the number and nature of its output parameters has three distinct areas:

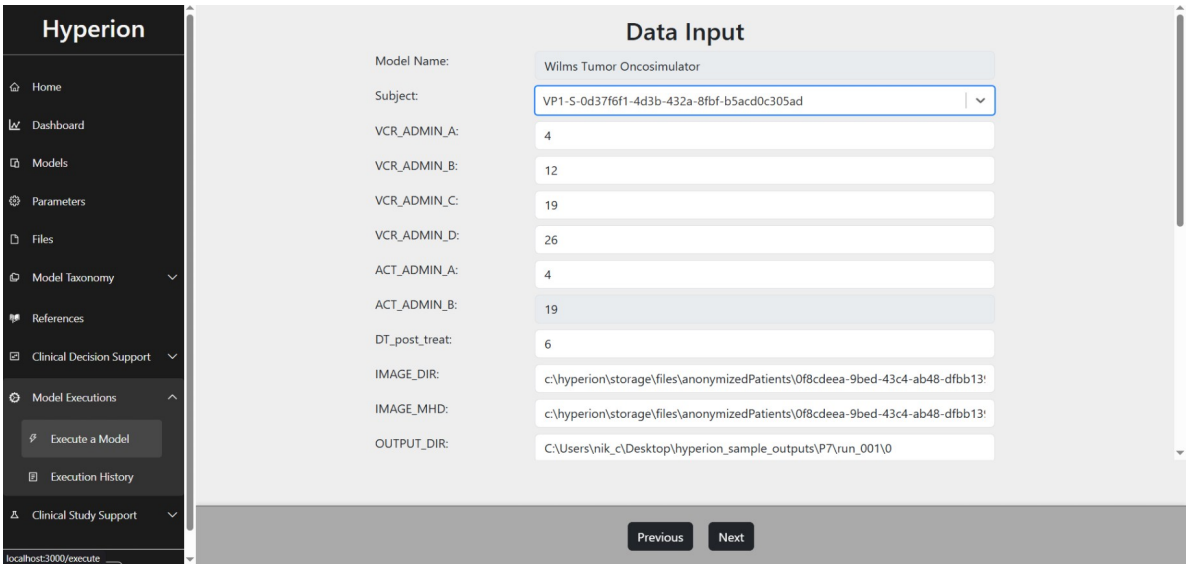


Figure 7-34: Model Execution Wizard data input screen for the Wilms Tumor Oncosimulator

- Output files: Usually reserved for graphical representations, this area presents any content is derived from files that cannot be processed to a higher granularity level (e.g. CSV files containing numerical data).
- Numerical results: In this part of the report, results taken either directly from variables or by files containing processable data are presented, using the stored information in the Model repository, such as parameter names and units to make them more presentable
- Clinical questions and answers: As described in section 6.4 for the definition of clinical questions and their answers using models and their parameters, the back end monitors each model execution when producing the values for its output parameters. Then it interferes and stores to the model execution results table the pertinent clinical question and the corresponding answer, depending on these values. Upon the creation of the report, the textual information from the Clinical Decision Support Repository is retrieved. This information is used to build a table with all the involved clinical questions and answers.

Finally, the produced report can be downloaded in PDF format by clicking the "Download PDF report" from within the Model Execution report screen, or the "Download" button of the desired row from the list screen.

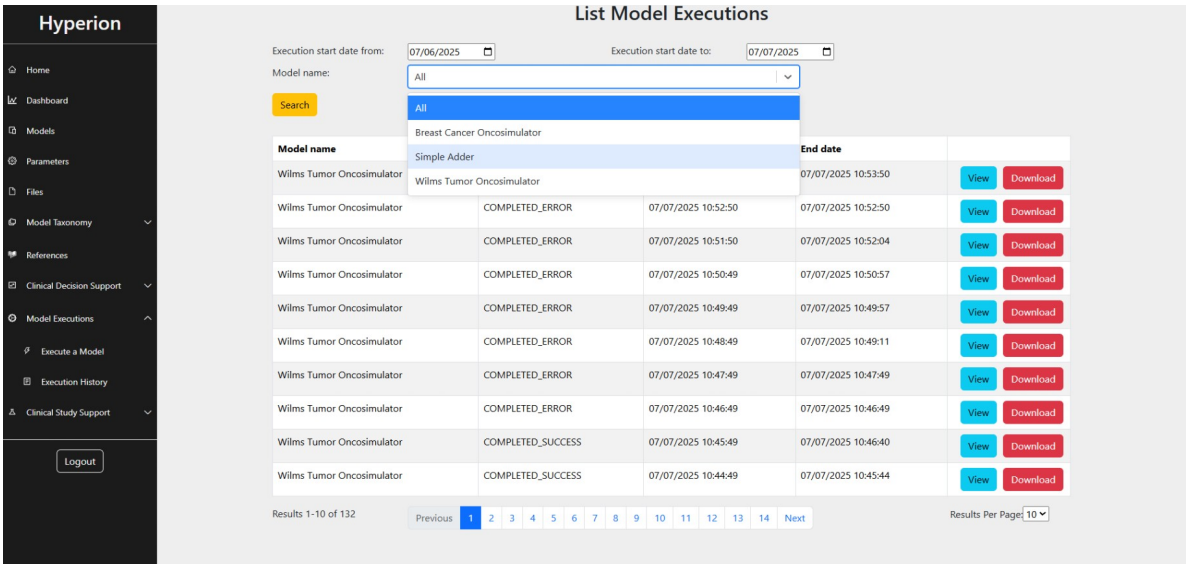


Figure 7-35: Model Execution list screen

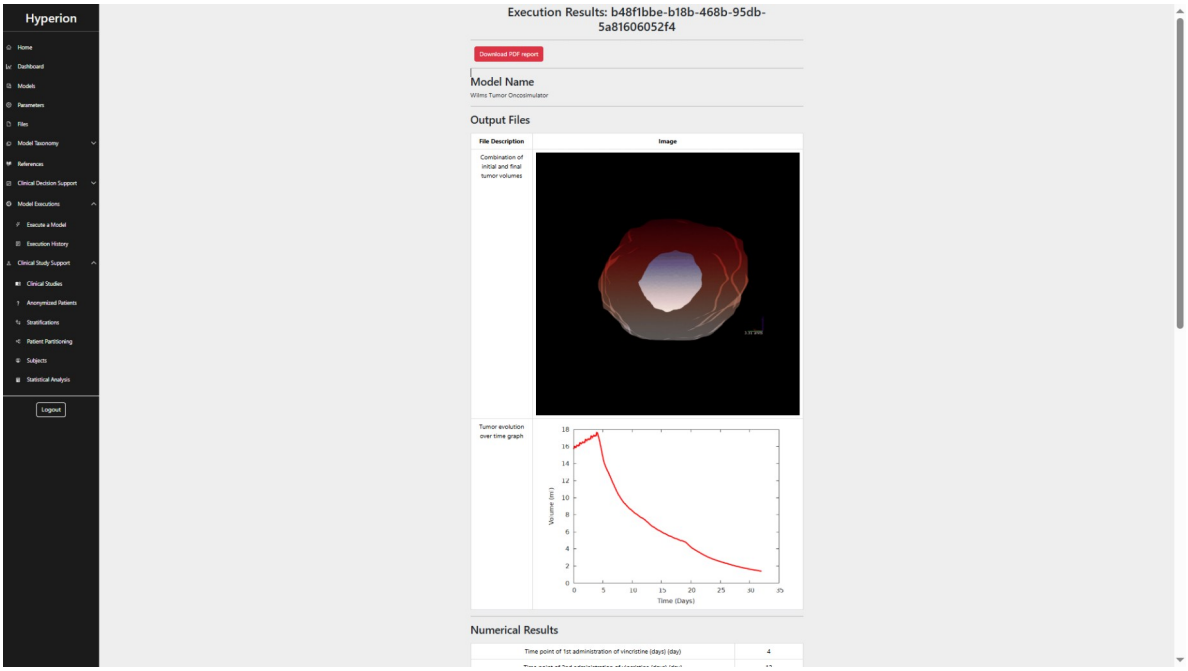


Figure 7-36: Model Execution report screen (part 1). The Output file area of the Wilms Tumor Oncosimulator contains a spinning image of the superimposed images of initial and final tumor volumes, as well as a tumor volume evolution graph over time.

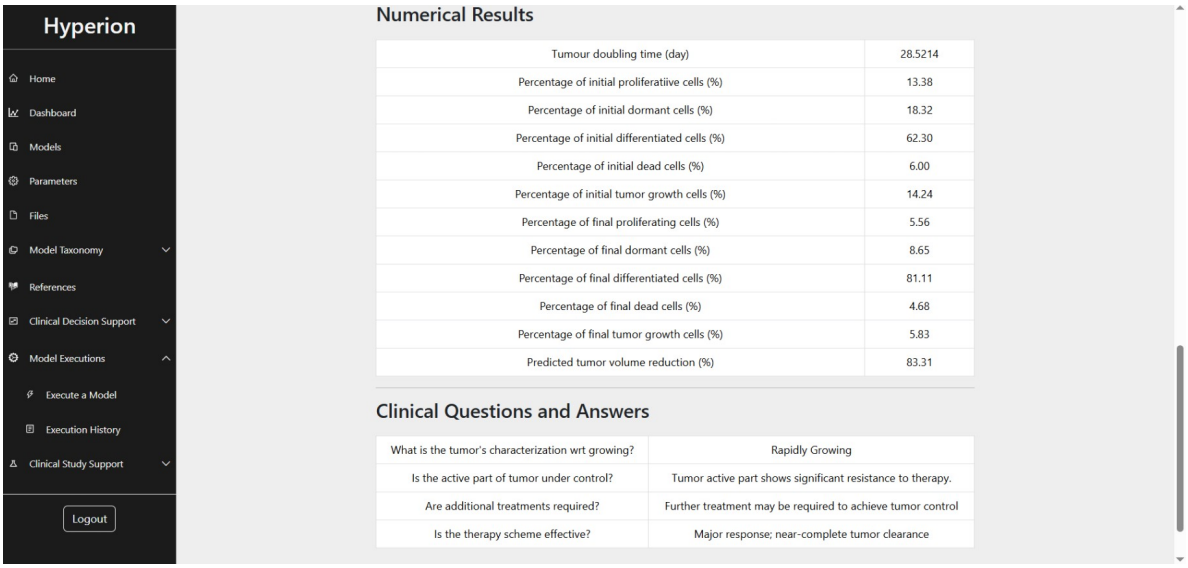


Figure 7-37: Model Execution report screen (part 2)

7.2.2 Clinical study support.

To further aid in result evaluation from a clinical study comprised of numerous model executions, graphical representations are required that will provide the relations between different inputs and outputs among the model's parameters, as data are collected and stored from the model executions conducted on Hyperion using the workflow described in the previous chapter. In order to utilize this functionality, the user can click on "Clinical Decision Support → Statistical Analysis" which will lead them to the corresponding screen.

To request statistical information about a clinical study, the user must choose values from the presented drop-downs in this succession: Model → Clinical Study → Stratification → Anonymized Patient. Choosing a drop-down value from one step will call the back end to populate the drop-down for the next one. Using the model and clinical study drop-downs are mandatory, since the manner in which the execution data are stored, in combination with the structure of the clinical study repository (chapters 6.3 and 6.4), means that data can be aggregated at a higher granularity level, that being an entire clinical study.

The input and output parameters that will be used to create the graphs depend on the type of graph that will be chosen. Currently three types of graphs are supported:

- Box Plot: This will produce statistical data regarding the collected values of a single input or output parameter among the various executions. One element will be produced per unit of the highest granularity chosen, be it the stratifications of a clinical study, or the anonymized patients of a stratification.
- Scatter Plot: This graph requires exactly 1 input and exactly 1 output parameter. It then produces a set containing the combinations (x,y) = (input_parameter_value, output_parameter_value) for all the executions that will be found under the highest granularity entity chosen. Each combination is a plotted point in the graph.
- Pair Plot: Extending the notion of a scatter plot, this choice allows the user to use as many input and output parameters. If X and Y be the respective amounts then this

will result in creating $(X+Y)^2$ scatter plots which are shown together in a rectangular grid format, ideally (X+Y) by (X+Y).

As an example the partial reproduction of the work in [1] is considered. Specifically, a single anonymized patient was entered into the system and a pertinent stratification was created. Then a number of executions were called using Hyperion's APIs. There was a set of initial input values for the Nephroblastoma Oncosimulator taken from bibliography and a chosen therapy schema defined by the administration time points (in days) of 4 Actinomycin and 2 Vincristine dosages.

The goal of the clinical study is to clinically adapt the Oncosimulator focusing of the total cell kill ratio parameter, composed of the cell kill ratio of the two administered drugs using the following equation set:

$$\mathbf{CKR_{Total}=CKR_{ACT}+CKR_{VCR}}$$
$$\mathbf{CKR_{ACT}=(3/5)*CKR_{Total}}$$
$$\mathbf{CKR_{VCR}=(2/5)*CKR_{Total}}$$

Therefore a set of executions were registered to Hyperion. These executions were done into separate runs. Each run followed this specific algorithm:

1) Start with $\mathbf{CKR_{ACT} = 0.3}$ and $\mathbf{CKR_{ACT} = 0.2.}$ This creates an initial value of 0.5 for $\mathbf{CKR_{Total}}$. Consider also an interval between 0 and 1.
2) Execute the model.
3) If the calculated tumor reduction percentage is not within 5% of the known reduction then set the new CKR = (lower interval limit + CKR)/2 or CKR = (upper interval limit + CKR)/2 depending if the calculated value is above, or below the known value, respectively
4) Repeat until calculated tumor reduction is within 5% of the known one.

In addition to the initial set of values, a number of subject groups was further considered. Each group was based on a slight deviation of only one of its other input parameters, randomly derived from the interval [0.5*parameter_value, 1.5*parameter_value]. 45 groups were formulated in total, resulting in 133 executions. The results of the pair plot for the parameters P_{sym}, P_{sleep}, CKR_{VCR}, CKR_{ACT} and T_d as=re given in figures 7-38 to 7-42.

For a single scatter plot and a box plot containing multiple elements the results are given in figures 7-43 and 7-44 respectively

Input Parameter	Description	Reference Value	Unit
$\mathbf{T_d}$	Doubling time of the tumor (size and cell population)	29	d
$\mathbf{T_c}$	Cell cycle duration	23	h
$\mathbf{T_{G0}}$	Time required for dormant cells to die through necrosis	96	h
$\mathbf{T_N}$	Time required for complete necrosis and removal of the products from the tumor	20	h

Input Parameter	Description	Reference Value	Unit
T_A	Time required for complete apoptosis and removal of the products from the tumor	6	h
R_A	% of undifferentiated cells that die by apoptosis per hour (STEM and LIMP)	0.001	1/h
$R_{A,Diff}$	% of differentiated cells that die by apoptosis per hour	0.003	1/h
$R_{N,Diff}$	% of differentiated cells that die by necrosis per hour	0.001	1/h
P_{G0toG1}	% of undifferentiated cells leaving the resting G_0 phase to re-enter the cell cycle (STEM and LIMP)	0.01	1/h
N_{LIMP}	Max number of mitoses a LIMP cell can undergo before terminal differentiation	3	—
P_{sym}	% of stem cells that divide symmetrically	0.45	1/h
P_{sleep}	% of cells entering G_0 phase after mitosis (STEM and LIMP)	0.28	1/h
CKR_{VCR}	Fraction of tumor cells affected by administered vincristine dose	0.3	—
CKR_{ACT}	Fraction of tumor cells affected by administered actinomycin dose	0.2	—
CKR_{Total}	Overall fraction of tumor cells affected by administered chemotherapy	0.5	—
VCR-ADMIN-A	Time of 1st vincristine administration	4	days
VCR-ADMIN-B	Time of 2nd vincristine administration	11	days
VCR-ADMIN-C	Time of 3rd vincristine administration	18	days
VCR-ADMIN-D	Time of 4th vincristine administration	25	days
ACT-ADMIN-A	Time of 1st actinomycin administration	4	days
ACT-ADMIN-B	Time of 2nd actinomycin administration	18	days
DT-post-treat	Duration between last drug administration and simulation end (MRI post-chemo)	1	days

Table 7-1: Initial input values of the Nephroblastoma Oncosimulator

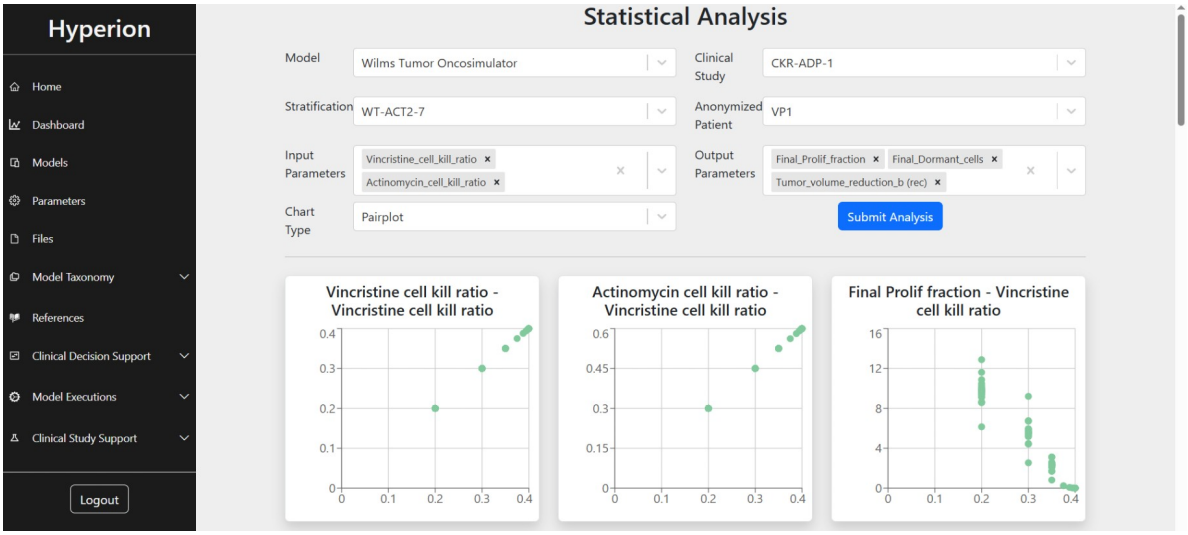


Figure 7-38: Pair plot results (part 1)

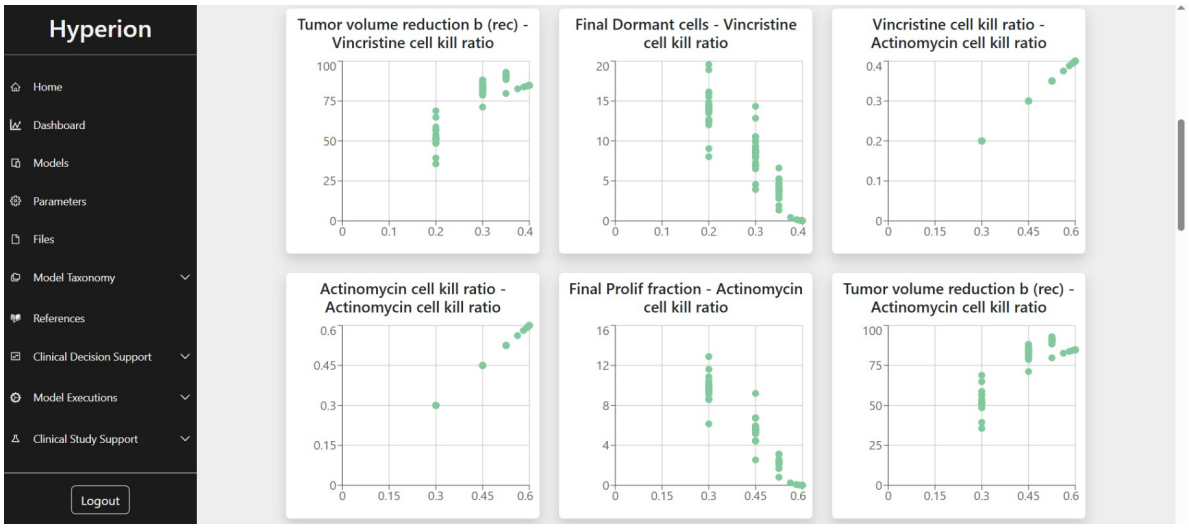


Figure 7-39: Pair plot results (part 2)

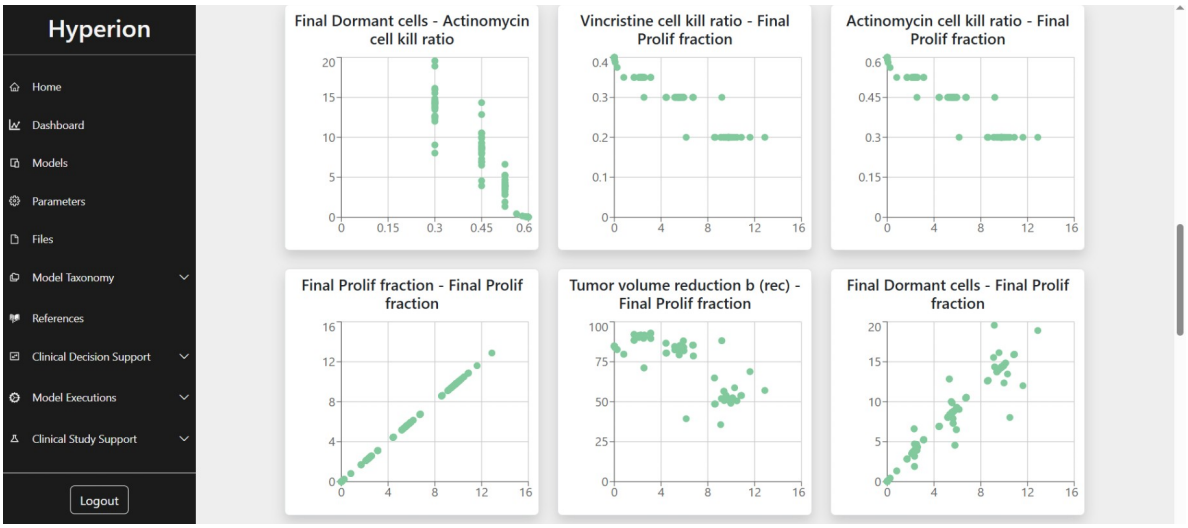


Figure 7-40: Pair plot results (part 3)

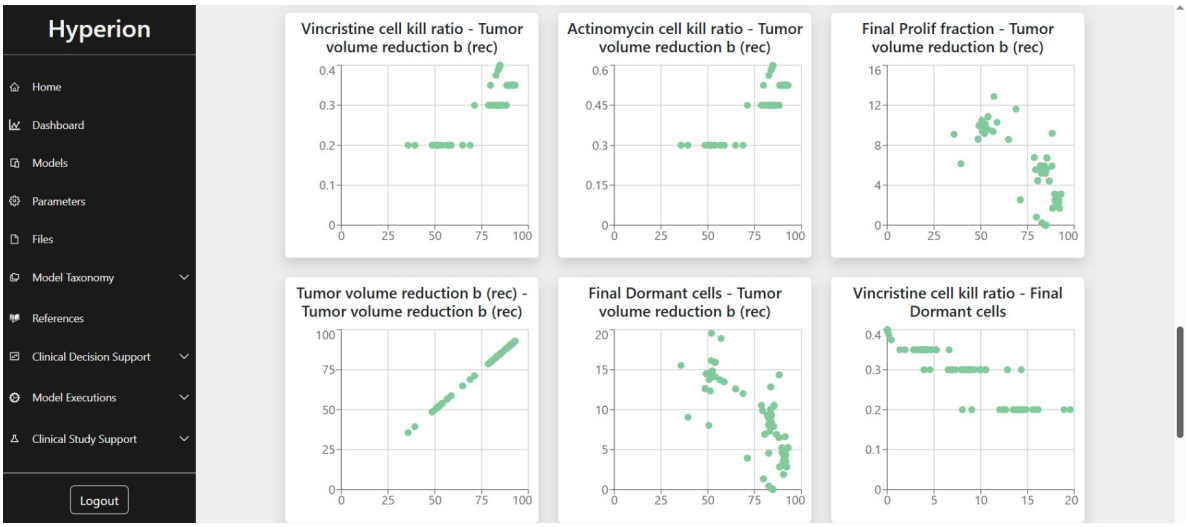


Figure 7-41: Pair plot results (part 4)

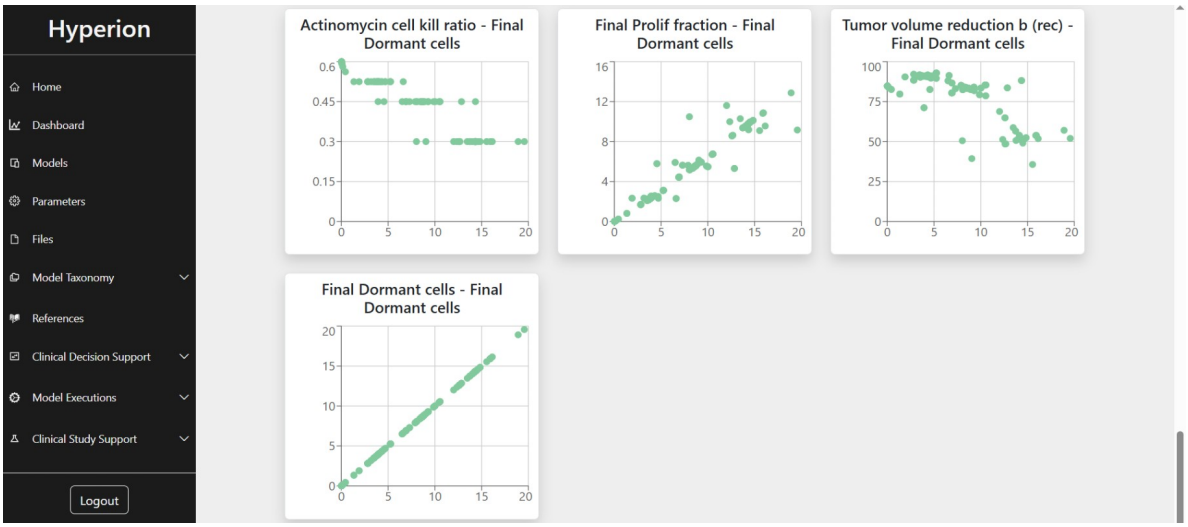


Figure 7-42: Pair plot results (part 5)

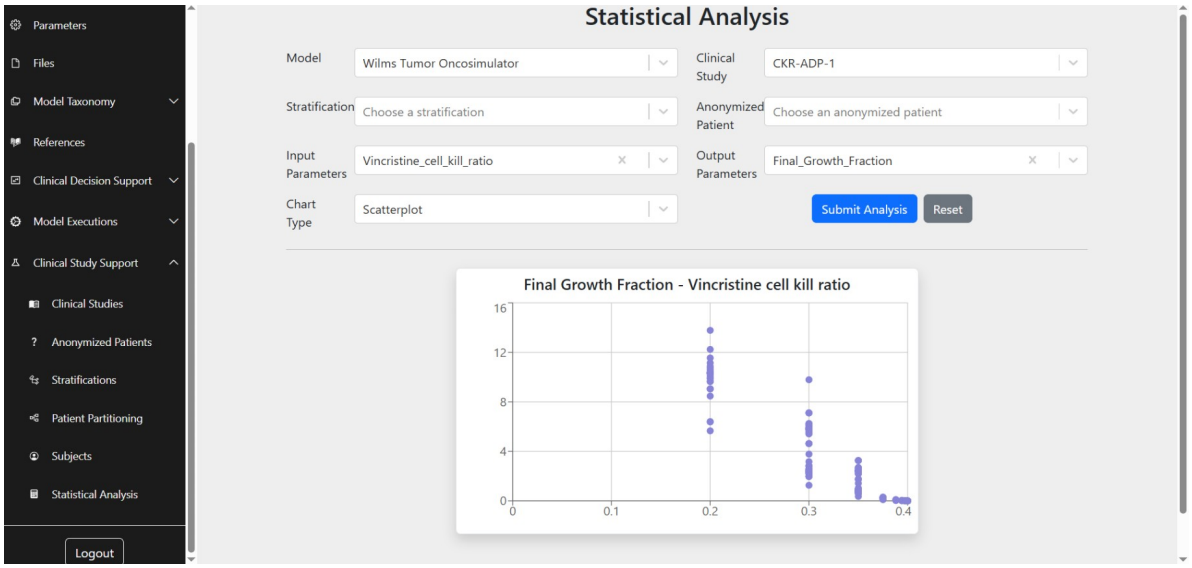


Figure 7-43: Scatter plot results for the tumor final growth fraction as per the different vincristine cell kill ratio values for the entire clinical study (no specific stratification or anonymized patient chosen)

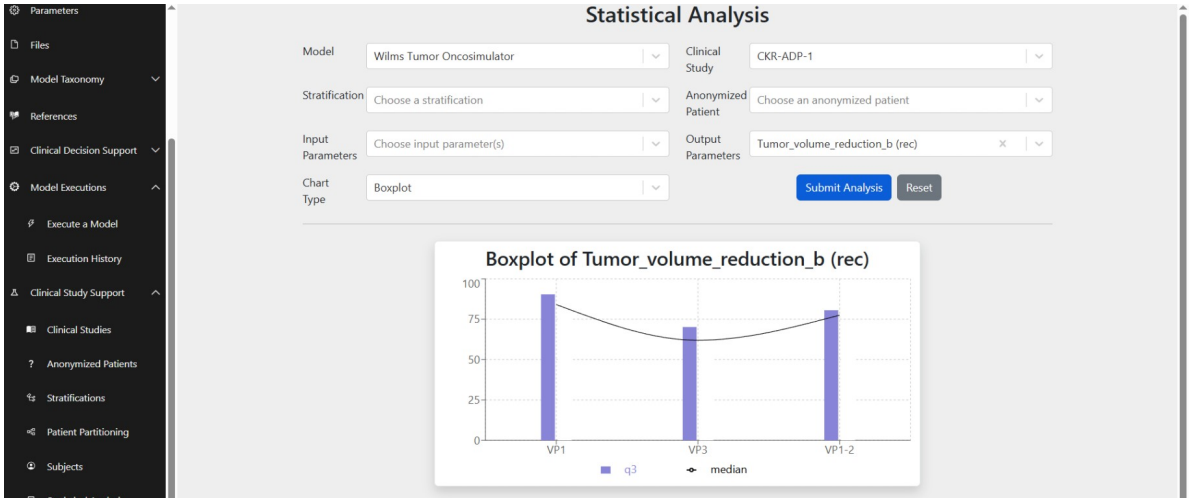


Figure 7-44: Box plot diagram of tumor volume reduction for all the executions pertaining to a set of patients included in a clinical study

7.3 References

1. Meyerheim M, Panagiotidou F, Georgiadi E, Soudris D, Stamatakos G and Graf N (2025) Exploring the *in silico* adaptation of the Nephroblastoma Oncosimulator to MRI scans, treatment data, and histological profiles of patients from different risk groups. *Front. Physiol.* 16:1465631. doi: 10.3389/fphys.2025.1465631

8. Conclusions

8.1 Discussion

Throughout this thesis, we tackled the difficulties of creating an infrastructure for hosting, and technologically integrating Oncosimulators, the end goal being to make them easier and more accessible and understandable to use for both clinical and research practice. We began from the conceptual foundations of personalized cancer simulation and progressed to the realization of organized, interoperable frameworks that can manage sophisticated models, execution data to the scale of a clinical study, and clinical questions.

By developing and deploying solid back end services, database schemas, and user interfaces, this project delivers an extensible and modular architecture to facilitate personalized simulation pipelines. This activity not only makes research more compliant with FAIR principles but also improves the collaboration among model developers and clinical users.

A central challenge was overcoming the gap from biological complexity to technical limitations. We addressed this by dividing concerns into distinct components: model hosting, features facilitating clinical studies, and integration with clinical decision support. The outcome is a system that is technically robust and clinically meaningful. Looking forward, future work may involve closer coupling with real-time EHR systems, greater explainability in simulation results, and extension to multi-disease platforms. Also, validation of the simulations via larger clinical trials and greater data integration is always a future aspiration. In summary, this thesis presents a first step toward rendering predictive simulation a practical, ethical, and effective instrument in personalized medicine. It establishes the foundations of in silico oncology systems that are intelligent but also accessible, reliable, and actionable.

8.2 Future expansions

While Hyperion in its current stage demonstrates a basic version of the capabilities and the benefits of a single-point simulation model management system, the potential of widespread use in the clinical practice does create a set of specific requirements. While meeting these requirements is theoretically possible from this version of the application, certain expansions need to be added to enhance the functionalities and thus its usability as a day-to-day clinical and research tool.

8.2.1 Optimization for handling multiple executions

To reduce the time needed for handling a large number of executions, for small scale applications, tools such as virtual thread handling or structured concurrency [1] are offered by Java to assist in handling tasks such as the one described in chapter 7.2.2. For large premises, such as a hospital, where theoretically the number of executions can reach up to thousands per day, proper IT infrastructures need to be in place, while Hyperion must have its API optimized to be able to communicate with such infrastructures. Usual examples include the use of tools such as distributed task queues (e.g., RabbitMQ [2], Apache Kafka

[3]), job orchestration frameworks (e.g., Kubernetes Jobs [4], Apache Airflow [5], Argo Workflows [6]), and containerization technologies like Docker [7] for packaging and isolating each execution unit. In such architectures, each execution can be submitted as an independent job, managed and scaled dynamically based on workload. Additionally, shared storage systems (e.g., NFS [8], S3 [9]) and centralized logging/monitoring platforms (e.g., Prometheus [10], Grafana [11]) are typically employed to ensure reliable data handling, traceability, and performance monitoring at scale.

8.2.2 Scalable Analysis and Reporting of Model Results

As the volume of model executions increases—particularly in high-throughput environments such as hospitals, pharmaceutical research facilities, or regulatory bodies—the volume, complexity, and heterogeneity of resulting data likewise expand. To enable comprehensive analysis, quality assurance, and longitudinal insights, the system must evolve beyond its initial output-handling mechanisms. This necessitates the implementation of ETL (Extract, Transform, Load) pipelines [12], which systematically extract relevant outcome metrics (e.g., tumor reduction percentages), transform them into normalized or denormalized formats, and load them into structured storage systems.

In practice, this often involves the creation of flattened analytical schemas within a separate data warehouse or analytical layer [13], decoupled from the transactional database used by the Hyperion application. These flattened schemas support efficient querying, aggregation, and visualization through business intelligence (BI) tools. For example, time-series analyses, cohort comparisons, and treatment effectiveness dashboards can be supported through pre-aggregated or indexed views.

In scenarios where the daily volume of results reaches the scale of millions of records, or where output data includes high-resolution time-series, genomic inputs, or imaging metadata, Big Data platforms become necessary to ensure scalability. Solutions such as Apache Spark [14], Hadoop HDFS [15], or cloud-native data warehouses like Amazon Redshift [16], Google BigQuery [17], and Azure Synapse Analytics [18] provide the parallel processing, fault tolerance, and distributed storage required for high-performance data analytics. These tools also facilitate integration with machine learning workflows, enabling predictive modeling and anomaly detection over large historical datasets.

Furthermore, a well-designed data processing layer enables automated reporting, regulatory audit trails, and real-time monitoring dashboards through integration with platforms such as Grafana or Power BI [19]. The combination of reliable data pipelines, scalable storage, and analytical computing forms a critical backbone for transforming raw model outputs into actionable clinical or operational insights.

8.2.3 Integration of Semantic Infrastructure into the Application Layer

To enable the semantic representation and querying of metadata derived from hypermodel executions, the system can be extended with a semantic data infrastructure as described in [20] and partially implemented in [21]. Apache Jena [22] is a mature, Java-

based framework that can be seamlessly embedded into the existing Spring Boot back end to support RDF data modeling, SPARQL querying, and ontology reasoning. Integration involves setting up a triplestore (e.g., Jena Fuseki [23]) to host domain-specific ontologies and execution metadata, along with a structured RDF generation pipeline from execution outputs. Optionally, inference capabilities can be added using Jena's built-in rule engine or OWL reasoners like HermiT [24], allowing enriched querying and knowledge extraction. This semantic layer facilitates advanced metadata discovery, supports federated queries across distributed knowledge bases, and aligns with the modular layout proposed in [20] for interoperability and extensibility in biomedical model management.

8.2.4 From Model-Based Clinical Questions to AI-Driven Hyper-Questions

In the current implementation of the Hyperion system, clinical hypotheses are typically formulated as targeted questions based on single or narrowly defined model parameters. Each question involves a specific parameter or a small set of predefined values (e.g., tumor volume threshold, drug concentration) and evaluates the outcome of a single model execution to determine an answer. While this provides deterministic, reproducible results and ensures interpretability, it limits the scope and depth of analysis in complex clinical scenarios where multiple variables interact and where combinatorial parameter spaces must be explored.

To overcome these limitations, a more scalable and intelligent system for formulating and answering clinical questions is required. This entails a paradigm shift from isolated, parameter-bound queries to what may be termed AI-assisted hyper-questions—questions that integrate multiple input parameters, models, and patient-specific features simultaneously. These hyper-questions can probe more complex hypotheses such as *"What treatment plan maximizes tumor regression across a specific genetic profile and comorbidity index within 6 months?"*—a task infeasible to address with single-parameter reasoning.

A critical enabler of this transition is the integration of artificial intelligence and machine learning frameworks into the existing simulation pipeline. Techniques such as Bayesian optimization [25], genetic algorithms [26], or reinforcement learning [27] can be employed to intelligently navigate large multidimensional parameter spaces and propose optimal value combinations. These AI agents can be trained to identify influential variables, propose novel parameter constellations, or prioritize combinations based on prior results, clinical constraints, or patient stratification.

In parallel, ensemble learning [28] or multi-model decision systems [29] can be incorporated to aggregate predictions from different simulation models (e.g., pharmacokinetics, cell population dynamics, immune response), enabling a unified response to hyper-questions that depend on multiple biological dimensions. This approach is particularly suited for hypermodels, as defined in [30], which combine different modeling techniques and data levels (molecular, tissue, patient-level) into a coordinated predictive framework.

The overall architecture to support hyper-question formulation requires not only AI algorithms but also an expressive query layer, ideally leveraging semantic technologies and ontologies to represent concepts, constraints, and model relationships [31]. This allows domain experts to express clinical hypotheses in human-readable terms that are internally mapped to computational queries over multiple models and datasets.mSuch an infrastructure enhances the scientific value and translational potential of the Hyperion platform. It enables automated hypothesis generation, adaptive model selection, and multi-scale optimization, making it suitable for real-time clinical decision support in precision oncology. Importantly, the system retains human-in-the-loop transparency, as AI-generated hypotheses or optimal combinations can always be traced back to underlying data and model rationale.

8.3 References

1. D. Beronić, P. Pufek, B. Mihaljević and A. Radovan, "On Analyzing Virtual Threads – a Structured Concurrency Model for Scalable Applications on the JVM", *2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO)*, Opatija, Croatia, 2021, pp. 1684-1689, doi: 10.23919/MIPRO52101.2021.9596855.

2. RabbitMQ: https://www.rabbitmq.com/

3. Apache Kafka: https://kafka.apache.org/

4. Kubernetes: https://kubernetes.io/

5. Apache Airflow: https://airflow.apache.org/

6. Argo Workflows: https://argo-workflows.readthedocs.io/en/latest/

7. Docker: https://www.docker.com/

8. T. Haynes and D. Noveck. 2015. RFC 7530: Network File System (NFS) Version 4 Protocol. RFC Editor, USA.

9. Amazon S3: https://aws.amazon.com/s3/

10. Prometheus: https://prometheus.io/docs/introduction/overview/

11. Grafana: https://grafana.com/

12. Kimball, R., & Caserta, J. (2004). *The data warehouse ETL toolkit*. John Wiley & Sons.

13. Inmon, W. H. (2005). *Building the data warehouse*. John wiley & sons.

14. Apache Spark: https://spark.apache.org

15. Apache Hadoop: https://hadoop.apache.org

16. Amazon Redshift: https://aws.amazon.com/redshift

17. Google BigQuery: https://cloud.google.com/bigquery

18. Microsoft Azure Synapse: https://azure.microsoft.com/en-us/products/synapse-analytics

19. Power BI: https://app.powerbi.com/

20. N. A. Christodoulou and G. S. Stamatakos, "A modular semantic infrastructure layout for the management of hypermodel-pertinent metadata in the context of In Silico oncology", *Proceedings of the 2014 6th International Advanced Research Workshop on In Silico Oncology and Cancer Investigation - The CHIC Project Workshop (IARWISOCI)*, Athens, Greece, 2014, pp. 1-4, doi: 10.1109/IARWISOCI.2014.7034640.

21. N. A. Christodoulou. (2012). *IntegraHEALTH 1.0: Application for Semantic Integration and Federated Search of Heterogeneous Healthcare Data* [Undergraduate diploma thesis, National Technical University of Athens]. DSpace NTUA. https://dspace.lib.ntua.gr/xmlui/handle/123456789/6107

22. Apache Jena: https://jena.apache.org

23. Apache Jena Fuseki: https://jena.apache.org/documentation/fuseki2/

24. Shearer, R. D., Motik, B., & Horrocks, I. (2008, October). Hermit: A highly-efficient OWL reasoner. In *Owled* (Vol. 432, p. 91).

25. Shahriari, B., et al. (2016). *Taking the Human Out of the Loop: A Review of Bayesian Optimization*. Proceedings of the IEEE.

26. Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press.

27. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.

28. Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. CRC Press.

29. Kuncheva, L. I. (2004). *Combining Pattern Classifiers: Methods and Algorithms*. Wiley.

30. Stamatakos, G. S., et al. (2010). *Multiscale Cancer Modeling and In Silico Oncology: Towards Integrative Clinical Trials*. Philosophical Transactions of the Royal Society A.

31. Whetzel, P. L., et al. (2011). *BioPortal: Enhanced functionality via new Web services from the National Center for Biomedical Ontology*. Nucleic Acids Research.

Appendix A: Hyperion database table structures

anonymized_patient		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(45)	Unique identifier of the anonymized patient
clinical_study_id	int	Id of the clinical study the patient is part of (Foreign Key to clinical_study table)
patient_id	varchar(1000)	Given id of the patient
age	int	Patient age in years
weight	int	Patient weight in kilos
height	int	Patient height in cm
gender	int	Patient gender
tumor_stage	varchar(1000)	Patient initial tumor stage
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

anonymized_patient_stratification		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the combination
anonymized_patient_id	int	Id of the anonymized patient (Foreign key to anonymized_patient table)
stratification_id	int	Id of the stratification (Foreign key to stratification table)
value	varchar(1000)	Combination value
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

clinical_question		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the clinical question
text	varchar(1000)	Clinical question text
description	varchar(1000)	Clinical question description
comment	varchar(1000)	Clinical question comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

clinical_question_answer		
Field	**Type**	**Description**
id	int	Primary key of the table

uuid	varchar(36)	Unique identifier of the answer
model_id	int	Id of the related model (Foreign key to model table)
clinical_question_id	int	Id of the clinical question that the answer relates to (Foreign key to clinical_question table)
parameter_id	int	Id of the parameter that the answer is based on (Foreign key to parameter table)
answer	varchar(1000)	Answer text
min_value	varchar(45)	Minimum parameter value for the answer
max_value	varchar(45)	Maximum parameter value for the answer
default_value	varchar(45)	Default parameter value for the answer
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

clinical_study		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the clinical study
model_id	int	Id of the model the clinical study relates to (Foreign key to model table)
name	varchar(1000)	Clinical study name
description	varchar(1000)	Clinical study description
comment	varchar(1000)	Clinical study comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

flyway_schema_history		
Field	**Type**	**Description**
installed_rank	int	A unique, incrementing number that represents the order in which the migration was applied.
version	varchar(50)	The version of the migration, as defined in the migration script filename (e.g., 1, 2.1, 3.4.5).
description	varchar(200)	Description of the migration, extracted from the script name (e.g., Create user table).
type	varchar(20)	Type of migration. Possible values: SQL, JDBC, UNDO, REPEATABLE.
script	varchar(1000)	The name of the migration script file that was applied (e.g., V1__create_user_table.sql).
checksum	int	Checksum calculated from the script contents to detect changes in migration files.

installed_by	varchar(100)	The database user that applied the migration.
installed_on	timestamp	The timestamp when the migration was applied.
execution_time	int	The time taken (in milliseconds) to execute the migration.
success	tinyint(1)	Indicates whether the migration was successfully applied (1 = success, 0 = failure).

model		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the model
name	varchar(1000)	Model name
description	varchar(1000)	Model description
comment	varchar(1000)	Model comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

model_execution		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the model execution
model_id	int	Id of the execution model (Foreign key to model table)
subject_id	int	Id of the execution subject (Foreign key to subject table)
status	int	Status of the model execution
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

model_execution_result		
Field	**Type**	**Description**
id	int	Primary key of the table
model_id	int	Id of the model (Foreign key to model table)
model_execution_id	int	Id of the model execution that contains the result (Foreign key to model_execution table)
parameter_id	int	Id of the result's parameter (Foreign key to parameter table)
parameter_value	varchar(1000)	Value of the result's parameter
clinical_question_id	int	Id of the clinical question related to result's parameter (Foreign key to clinical_question table)
clinical_question_answer	int	Id of the answer of the clinical question

_id		based on the parameter's value (Foreign key to clinical_question_answer table)
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

model_file		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the file
model_id	int	Id of the model the file refers to (Foreign key to model table)
name	varchar(255)	File name
description	varchar(1000)	File description
type	int	File type
source	varchar(255)	Location of the source file within the file system
license	varchar(1000)	Licence of the file (applicable for code/executables)
checksum	varchar(255)	File checksum
architecture	int	Build architecture of the file (applicable for executables)
comment	varchar(1000)	File comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

model_property		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the combination
model_id	int	Id of the combination's model (Foreign key to model table)
property_id	int	Id of the combination's property (Foreign key to property table)
value	varchar(1000)	Combination's value
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

parameter		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the parameter
model_id	int	Id of the model the parameter is part of (Foreign key to model table)
name	varchar(1000)	Parameter name

description	varchar(1000)	Parameter description
container_parameter_id	int	Id of the parameter that contains the parameter (applicable for files that contain parameters values – Self reference foreign key)
argument_flag	varchar(1000)	Parameter argument flag used for the model execution statement
data_type	int	Parameter data type
unit	varchar(45)	Parameter unit
min_value	varchar(45)	Parameter minimum value
max_value	varchar(45)	Parameter maximum value
default_value	varchar(45)	Parameter default value
is_output	tinyint(1)	Parameter characterization input/output
semantic_url	varchar(1000)	Semantic url of the parameter
comment	varchar(1000)	Parameter comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

property		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the property
name	varchar(1000)	Property name
description	varchar(1000)	Property description
semantic_url	varchar(1000)	Semantic url of the property
comment	varchar(1000)	Property comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

reference		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the reference
model_id	int	Id of the reference's model (Foreign key to model table)
title	varchar(1000)	Reference title
type	varchar(1000)	Reference type
author	varchar(1000)	Reference author
issued	date	Reference issue date
citation	varchar(1000)	Reference citation
doi	varchar(1000)	Reference doi
pmid	varchar(1000)	Reference pmid
log_date	timestamp	Date and time of first log entry

last_edited	timestamp	Date and time of last entry edit

stratification		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the stratification
name	varchar(1000)	Stratification name
model_id	int	Id of the related model (Foreign key to model table)
parameter_id	int	Id of the related parameter (Foreign key to parameter table)
default_value	varchar(1000)	Default parameter value of the stratification
min_value	varchar(1000)	Minimum parameter value of the stratification
max_value	varchar(1000)	Maximum parameter value of the stratification
comment	varchar(1000)	Stratification comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

subject		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the subject
subject_id	varchar(1000)	Given Id of the subject
anonymized_patient_id	int	Id of the subject's anonymized patient (Foreign key to anonymized_patient table)
stratification_id	int	Id of the subject's stratification (Foreign key to stratification table)
comment	varchar(1000)	Comment for the subject
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

subject_file		
Field	**Type**	**Description**
id	int	Primary key of the table
uuid	varchar(36)	Unique identifier of the subject file
subject_id	int	Id of the combination's subject (Foreign key to subject table)
parameter_id	int	Id of the combination's parameter (Foreign Key to parameter table)
name	varchar(255)	File name
description	varchar(1000)	File description
type	int	File type
source	varchar(255)	Location of the source file within the file system

checksum	varchar(255)	File checksum
comment	varchar(1000)	File comment
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

subject_parameter		
Field	**Type**	**Description**
id	int	Primary key of the table
subject_id	int	Id of the combination's subject (Foreign Key to subject table)
parameter_id	int	Id of the combination's parameter (Foreign Key to parameter table)
value	varchar(100)	Combination's value
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

user		
Field	**Type**	**Description**
id	int	Primary key of the table
username	varchar(255)	Username of the user
password	varchar(255)	Password of the user (encrypted)
role	varchar(255)	User role in the system
log_date	timestamp	Date and time of first log entry
last_edited	timestamp	Date and time of last entry edit

Appendix B: List of Abbreviations

ACGT – Advancing Clinico-Genomic Trials on Cancer
ACID – Atomicity, Consistency, Isolation, Durability
AI – Artificial Intelligence
API – Application Programming Interface
BI – Business Intelligence
CDISC – Clinical Data Interchange Standards Consortium
CDM – Common Data Model
CDSS – Clinical Decision Support System
CHIC – Computational Horizons In Cancer
CSV – Comma-Separated Values
CPOE – Computerized Provider Order Entry
ContraCancrum – Clinically Oriented Translational Cancer Multilevel Modelling
CRISPR – Clustered Regularly Interspaced Short Palindromic Repeats
CRUD – Create, Read, Update, Delete
CSCO AI – Chinese Society of Clinical Oncology Artificial Intelligence
DDD – Domain-Driven Design
DMS – Data Management System
DOM – Document Object Model
EHR – Electronic Health Record
ER – Entity Relationship
ETL – Extract-Transform-Load
FAIR – Findability, Accessibility, Interoperability, Reusability
FP7 – Seventh Framework Programme of the European Community for research and technological development and demonstration activities
GDPR – General Data Protection Regulation
GPOH – Clinical Trials in Paediatric Oncology and Haematology
GUI – Graphical User Interface
HDFS – Hadoop Distributed File System
HIPAA – Health Insurance Portability and Accountability Act
HPC – High Performance Computing
HTTP – Hypertext Transfer Protocol
HTTPS – Hypertext Transfer Protocol Secure
ICCS – Institute of Communication and Computer Systems
IEC – International Electrotechnical Commission
ISO – International Organization for Standardization
JSON – JavaScript Object Notation
JWT – JSON Web Token
LOINC – Logical Observation Identifiers Names and Codes
MCR – MATLAB Compiler Runtime
MD – Doctor of Medicine
ML – Machine Learning
MRI – Magnetic Resonance Imaging
MTV – Model–Template–View
MVC – Model–View–Controller
NCCN – National Comprehensive Cancer Network
NFS– Network File System

NIH – National Institutes of Health
NTUA – National Technical University of Athens
OMOP – Observational Medical Outcomes Partnership
ORM – Object–Relational Mapping
OS – Operating System
OWL – Web Ontology Language
PDF – Portable Document Format
PET – Positron Emission Tomography
PICO – Patient, Intervention, Comparison, Outcome
PICOT – Patient, Intervention, Comparison, Outcome, Timeframe
RDBMS – Relational Database Management System
RDF – Resource Description Framework
REST – Representational State Transfer
SIOP – International Society of Pediatric Oncology
SNOMED CT – Systematized Nomenclature of Medicine – Clinical Terms
SPARQL – SPARQL Protocol and RDF Query Language
SPARQL-DQP – SPARQL Distributed Query Processing
UI – User Interface
URL – Uniform Resource Locator
UUID – Universally Unique Identifier
VEGF – Vascular Endothelial Growth Factor
VPN – Virtual Private Network
VPHi – Virtual Physiological Human Institute
WFO – Watson for Oncology
XML – Extensible Markup Language

ΕΜΠ – Εθνικό Μετσόβιο Πολυτεχνείο
ΕΠΙΣΕΥ – Ερευνητικού Πανεπιστημιακού Ινστιτούτου Συστημάτων Επικοινωνιών και Υπολογιστών

Appendix C: Publications

1. Publications in peer-reviewed scientific journals:

- Christodoulou NA, Tousert NE, Georgiadi EC, Argyri KD, Misichroni FD, Stamatakos GS. A Modular Repository-based Infrastructure for Simulation Model Storage and Execution Support in the Context of In Silico Oncology and In Silico Medicine. Cancer Inform. 2016 Oct 27;15:219-235. doi: 10.4137/CIN.S40189. PMID: 27812280; PMCID: PMC5084707.
- Bucur A, van Leeuwen J, Christodoulou N, Sigdel K, Argyri K, Koumakis L, Graf N, Stamatakos G. Workflow-driven clinical decision support for personalized oncology. BMC Med Inform Decis Mak. 2016 Jul 21;16 Suppl 2(Suppl 2):87. doi: 10.1186/s12911-016-0314-3. PMID: 27460182; PMCID: PMC4965727.

2. Publications in conference proceedings:

- Eleni Ch. Georgiadi, Nikolaos A. Christodoulou, Christos Kyroudis, Feng Dong, Norbert Graf, and Georgios S. Stamatakos. Oncosimulator Models as Components of a Personal Health Record Platform can Enable and Enhance the Provision of Personalized Medical Treatment. VPH2016, book of abstracts, University of Amsterdam, 297 - 300 , ISBN 978-90-826254-0-0
- N. A. Christodoulou and G. S. Stamatakos, "A modular semantic infrastructure layout for the management of hypermodel-pertinent metadata in the context of In Silico oncology," *Proceedings of the 2014 6th International Advanced Research Workshop on In Silico Oncology and Cancer Investigation - The CHIC Project Workshop (IARWISOCI)*, Athens, Greece, 2014, pp. 1-4, doi: 10.1109/IARWISOCI.2014.7034640.